

UCC 5G cnSGWc Configuration and Administration Guide, Release 2026.04

First Published: 2026-11-05

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2026 Cisco Systems, Inc. All rights reserved.



CONTENTS

PREFACE

About this Guide xxxv

Conventions Used xxxv

Contacting Customer Support xxxvi

CHAPTER 1

5G Architecture 1

Architectural Overview 1

Control Plane Network Functions 1

User Plane Network Function 2

Subscriber Microservices Infrastructure Architecture 2

Control Plane Network Function Architecture 4

CHAPTER 2

cnSGW-C Overview 7

Product Description 7

Converged Core Overview 7

Converged core architecture and ULB integration 8

ULB integration in converged core 9

ULB integration design 9

ULB integration of GTPC ingress/egress traffic 10

ULB integration of PFCP ingress/egress traffic 11

Use Cases 12

Deployment Architecture and Interfaces 15

cnSGW-C Architecture 16

cnSGW-C Deployment 16

Converged Core Architecture 18

Converged Core Deployment 19

Supported Interfaces 20

Life Cycle of Data Packet 20
License Information 20
Standards Compliance 20

CHAPTER 3 Deploying and Configuring cnSGW-C through Operations Center 21
Feature Summary and Revision History 21
Summary Data 21
Revision History 21
Feature Description 22
cnSGW-C Ops Center 22
Prerequisites 22
cnSGW-C Service Configuration 23
Mapping Pods with Node Labels 23
Deploying and Accessing cnSGW-C 24
Deploying cnSGW-C 24
Accessing the cnSGW-C Ops Center 24
Day 0 Configuration 24
Loading Day 1 Configuration 27
Day1config.cli 27

CHAPTER 4 cnSGW-C Rolling Software Update 45
Feature Summary and Revision History 45
Summary Data 45
Revision History 45
Introduction 45
Updating cnSGW-C 46
Rolling Software Update Using the SMI Cluster Manager 47
Prerequisites 48
Triggering the Rolling Software Upgrade 52
Monitoring the Update Procedure 53
Viewing the Pod Details 54
Rolling Software Update on Non-SMI Cluster 56
Rolling Upgrade Optimization 57
Feature Description 59

How Rolling Upgrade Optimization Works 59
Pod Upgrades 60
Upgrading Software to Version with Rolling Upgrade Optimization Support 61
Limitations 62
Configuring the Supported Features for Rolling Upgrade 63
Verifying Rolling Upgrade Optimization 63
OAM Support 64
Bulk Statistics 64

CHAPTER 5 **Pods and Services Reference** 65
Feature Summary and Revision History 65
Summary Data 65
Revision History 65
Feature Description 65
Pods 66
Service pod 68
UDP Proxy Pod 68
Feature Description 68
Services 70
Open Ports and Services 72
Associating Pods to the Nodes 73
Viewing the Pod Details and Status 74
Pod Details 74
States 75
Runtime pod scales for server additions 75
Addition and scaling of runtime servers in converged core process 76

CHAPTER 6 **3GPP RAN/NAS Cause Codes Support** 79
Feature Summary and Revision History 79
Summary Data 79
Revision History 79
Feature Description 79
How it Works 81
Call Flows 81

Create Bearer Procedure Call Flow 81
Update Bearer Procedure Call Flow 82
Delete Bearer Command Procedure Call Flow 83
Delete Session Procedure Call Flow 84

CHAPTER 7 **Access Bearer Release Support 87**
Feature Summary and Revision History 87
Summary Data 87
Revision History 87
Feature Description 87
How it Works 88
Call Flows 88
Release Access Bearer (Active to IDLE Transaction) Call Flow 88

CHAPTER 8 **APN Profile Support 91**
Feature Summary and Revision History 91
Summary Data 91
Revision History 91
Feature Description 92
Feature Configuration 92
Configuring DNN Profile 92
Configuring Network Element Profile 92
Configuration Modification Impact 93
Validation of Uplink Packets for IP Source Violation 94
How Validation of Uplink Packets for IP Source Violation Works 95
Enable and Disable Validation of Uplink Packets for IP Source Violation 96
Verify Uplink Packet Source Validation on DNN Profile 97
Verify Uplink Packet Source Validation for NF Service 97
Troubleshooting Information 98
Configuration Errors 98

CHAPTER 9 **Change Notification Request Handling 101**
Feature Summary and Revision History 101
Summary Data 101

Revision History 101
Feature Description 101
Standards Compliance 102
How it Works 102
Call Flows 102
Change Notification Request Call Flow 102
OAM Support 104
Bulk Statistics Support 104

CHAPTER 10 **Clear Subscriber Request** 107
Feature Summary and Revision History 107
Summary Data 107
Revision History 107
Feature Description 107
Standards Compliance 108
How it Works 108
Supported Clear Command 109
Call Flows 109
Clear PDN Call Flow 109

CHAPTER 11 **Context Replacement Support** 113
Feature Summary and Revision History 113
Summary Data 113
Revision History 113
Feature Description 114
How it Works 114
Call Flows 114
Full Context Replacement Call Flow 114
Partial Context Replacement Call Flow 115
OAM Support 119
Bulk Statistics 119

CHAPTER 12 **Dedicated Bearer Support** 121
Feature Summary and Revision History 121

Summary Data 121
Revision History 121
Feature Description 121
Setup and Update Dedicated Bearers 122
Feature Description 122
How it Works 122
Call Flows 122
Delete Dedicated Bearers 129
Feature Description 129
How it Works 129
Call Flows 129

CHAPTER 13 **Delete Bearer and Delete Session Request** 133
Feature Summary and Revision History 133
Summary Data 133
Revision History 133
Feature Description 133
Delete from MME 134
Delete from PGW 134
Standard Compliance 134
How it Works 134
Call Flows 134

CHAPTER 14 **Downlink Data Notification** 139
Feature Summary and Revision History 139
Summary Data 139
Revision History 139
Feature Description 140
DDN Message Handling 140
Feature Description 140
How it Works 140
Call Flows 140
Feature Configuration 146
Configuring the DDN Failure Timer 147

Configuring DDN No User Connect Retry Timer 147
Configuration Example 148
Configuration Verification 148
Control Messages Triggered DDN Support 148
Feature Description 148
How it Works 148
Call Flows 148
Feature Configuration 150
Configuration Example 150
Configuration Verification 150
Disabling the DDN Control Procedure 150
DDN Advance Features 150
Feature Description 150
How it Works 151
Call Flows 151
Standards Compliance 157
Feature Configuration 157
Configuration Example 158
OAM Support 158
Bulk Statistics 158

CHAPTER 15 **DSCP Marking Support** 161
Feature Summary and Revision History 161
Summary Data 161
Revision History 161
Feature Description 162
DSCP Marking for Data Packets 162
Feature Description 162
How it Works 162
Feature Configuration 162
Configuration Example 164
Configuration Verification 164
DSCP Marking for CP Signaling Messages 164
Feature Description 164

Feature Configuration 164
Configuring DSCP under S11 Interface for GTP Endpoint 165
Configuring DSCP under S5e Interface for GTP Endpoint 165
Configuring DSCP under Sxa Interface for Protocol Endpoint 166
Removing DSCP Configuration 166

CHAPTER 16 **Dynamic Routing by Using BGP** 169
Feature Summary and Revision History 169
Summary Data 169
Revision History 169
Feature Description 170
How it Works 170
External Network Failure 172
Geo Switchover 172
Internal Network Failure 173
Local Switchover 173
Recovery and Failback 173
Call Flows 174
Publish Route for Incoming Traffic in an Active-Standby Mode 174
Single Protocol Pod Failure Call Flow 176
Learn Route for Outgoing Traffic Call Flow 176
Configuring Dynamic Routing Using BGP 177
Monitoring and Troubleshooting 180

CHAPTER 17 **Emergency Call Support** 185
Feature Summary and Revision History 185
Summary Data 185
Revision History 185
Feature Description 185
Limitations 186
How it Works 186
Call Flows 186
Create Emergency Session Call Flow 186
OAM Support 188

Bulk Statistics Support 188

CHAPTER 18 **Enhanced PFCP Association Release Procedure for Graceful Session Termination** 189

Call Disconnection Notification from UPF to cnSGWc through PFCP Association Release 190

UPF-initiated PFCP Session Release 191

UPF-initiated Enhanced PFCP Association Release 192

Enabling EPFAR to Initiate the PFCP Session Release Procedure 192

Bulk Statistics 193

CHAPTER 19 **Extended and Non-Standard QCI Values Support and Validation** 197

Feature Summary and Revision History 197

Summary Data 197

Revision History 197

Feature Description 197

Validation for Extended and Non-Standard QCI Values 198

Support and Validation for Extended and Non-Standard QCI Values for VoLTE Marking 198

CHAPTER 20 **eMPS and WPS Support** 199

Feature Summary and Revision History 199

Summary Data 199

Revision History 199

Feature Description 199

eMPS/WPS Support 200

Feature Description 200

eMPS GTPv2 Load/Overload Self Protection Exclusion Support 200

Feature Description 200

Message Priority Profile 201

Feature Description 202

How Message Priority Profile Selection Works 203

Feature Configuration 205

Configuring WPS Profile 205

Configuration Example 206

Configuration Verification 206

Configuring Message Priority Profiles 206

Configuring WPS-Profile and SGW-Profile Association 207
Configuration Example 207
Configuration Verification 207
Configuring WPS-Profile and DNN-Profile Association 207
Configuration Example 208
Configuration Verification 208
Configuring SGW QoS Profile 208
Associating sgw-qos-profile with sgw-profile and DNN profile 209
OAM Support 209
Monitoring and Troubleshooting 209
Bulk Statistics Support 211

CHAPTER 21 **Failure and Error Handling Support** 213
Feature Summary and Revision History 213
Summary Data 213
Revision History 213
Overview 214
Attach and Detach Failure and Error Handling 214
Create Session Request Failure Handling 214
Delete Default Bearer Procedure Failure Handling 215
Delete Session Procedure Failure Handling 216
Session Setup Timer during Attach Procedure 216
Create-Update-Delete Bearer Request and Response Failure and Error Handling 217
Create Bearer Procedure Failure Handling 217
Delete Dedicated Bearer Procedure Failure Handling 218
Update Bearer Procedure Failure Handling 219
Radio Access Bearer/Modify Bearer Request Failure and Error Handling 222
Support for Failure Cause Code, Cause Source, and Bearer Context Error 224
Failure Cause Code 224
Cause Source 224
Bearer Context Error 224

CHAPTER 22 **GTPC and Sx Path Management** 225
Feature Summary and Revision History 225

Summary Data 225
Revision History 226
Feature Description 226
GTPC and Sx Path Management 226
Feature Description 226
Feature Configuration 227
Configuring the Echo Parameters 227
Configuring Heartbeat 227
Viewing the Peer Configuration 228
Configuration Example 229
Limitations 229
OAM Support 229
Alerts 230
Bulk Statistics Support 230
Uniformity in compression at N4 and Sx interfaces 231
GTPC Path Failure 232
Feature Description 232
How it Works 233
GTPC Path Failure Detection 233
Path Failure Handling 233
Feature Configuration 234
Configuring Action on Path Failure Detection 234
Configuring Notification to Update the Peer Node 234
Configuration Example 234
OAM Support 235
Bulk Statistics Support 235
Sx Path Failure 235
Feature Description 235
How it Works 235
Sx Path Failure Detection 236
Path Failure Handling 236
Heartbeat Handling 236
OAM Support 237
Bulk Statistics Support 237

Customization of Path Failure Detection 237
Feature Description 237
Feature Configuration 238
Configuring Sx Path Failure Customization 238
Configuring GTPC Path Failure Customization 238
OAM Support 239
Bulk Statistics Support 239

CHAPTER 23 GTPU Error Indication 243
Feature Summary and Revision History 243
Summary Data 243
Revision History 243
Feature Description 244
How it Works 244
Error Indication Support 244
Default Bearer with s1u as local-purge Call Flow 245
Dedicated Bearer with s1u as local-purge Call Flow 247
Dedicated Bearer (IDFT) with s1u as local-purge Call Flow 248
Default/Dedicated Bearer with s1u as page-ue Call Flow 250
Default Bearer with s5u as local-purge/signal-peer Call Flow 251
Dedicated Bearer with s5u as local-purge/signal-peer Call Flow 253
Graceful Termination 255
Graceful Termination Call Flow 255
Session Replacement 257
Session Replacement for Default Bearer Call Flow 257
Session Replacement for Dedicated Bearer Call Flow 259
Feature Configuration 261
Configuration Example 261
Conifguration Verification 262
OAM Support 262
Bulk Statistics 262

CHAPTER 24 GTPU Path Failure 265
Feature Summary and Revision History 265

Summary Data 265
Revision History 265
Feature Description 266
How it Works 266
Call Flows 267
Path Failure for Default Bearer Call Flow 267
Path Failure for Dedicated Bearer Call Flow 268
Feature Configuration 270
Configuration Example 270
Configuration Verification 271
GTPU Path Failure OAM Support 271
Bulk Statistics 271

CHAPTER 25 **GTPv2 and Sx Messages Retransmission and Timeout Handling** 273
Feature Summary and Revision History 273
Summary Data 273
Revision History 273
Feature Description 274
How it Works 274
Configuring the Retransmission and Timeout Values 275
Configuration Verification 276

CHAPTER 26 **GTPv2 Load/Overload Support** 279
Feature Summary and Revision History 279
Summary Data 279
Revision History 279
Feature Description 279
Configuring the GTPv2 Load and Overload Feature 281
Configuring the Load Profile 281
Configuration Example 282
Configure the Overload Exclude Profile 282
Configuring the Overload Condition Profile 283
Configuring the Maximum Session Count 285
Configuration Example 285

Associating the Overload-Profile with SGW-Profile Association 285
Configuration Example 287
Configuration Verification 288
Configure Load Factor Parameters 288
Configuration Verification 289
GTPv2 Load and Overload OAM Support 289
Bulk Statistics 289

CHAPTER 27 GTPv2 Message Validation 291
Feature Summary and Revision History 291
Summary Data 291
Revision History 291
Feature Description 291
How it Works 292
Call Flows 292
Basic and Advance Validation on SGW-Ingress (S11) Call Flow 292
Basic and Advance Validation on SGW-Egress (S5) Call Flow 294

CHAPTER 28 IDFT Support 297
Feature Summary and Revision History 297
Summary Data 297
Revision History 297
Feature Description 297
Standards Compliance 298
How it Works 298
Call Flows 298
IDFT Support without SGW Relocation Call Flow 298
IDFT Support with SGW Relocation Call Flow 300
5G to 4G Handover Flow for Pure-S Call Flow 301
4G to 5G Handover Flow for Pure-S Call Flow 303
Create IDFT (System-level) Call Flow 305
Delete IDFT (System-level) Call Flow 307
OAM Support 308
Viewing IDFT Configuration 308

Failure Handling 310
Bulk Statistics Support 312

CHAPTER 29 Idle Session Timeout Settings 313
Feature Summary and Revision History 313
Summary Data 313
Revision History 313
Feature Description 313
How it Works 314
Call Flows 314
Inactivity Report Call Flow 314
Idle Timer Handling on UPF Call Flow 316
Reactivity Report Call Flow 318
Clear Call Handling Call Flow 319
Feature Configuration 320
Configuration Example 320
Configuration Verification 320

CHAPTER 30 Initial Attach Support 321
Feature Summary and Revision History 321
Summary Data 321
Revision History 321
Feature Description 322
How it Works 322
Call Flows 322
Initial Attach Call Flow 322
Standards Compliance 325
Support for Backoff Timer, Origination TimeStamp, and MaxWait Time 325
Backoff Timer 325
Origination Time Stamp 325
MaxWaitTime 326

CHAPTER 31 Inter System RAT Handover 327
Feature Summary and Revision History 327

Summary Data 327
Revision History 327
Feature Description 327
How it Works 328
Call Flows 328
Wi-Fi to LTE Success Call Flow 328
GnGp to LTE Handover with OI Indicator Set Call Flow 330
GnGp to LTE Handover with OI Indicator Unset Call Flow 331
Standards Compliance 333

CHAPTER 32 **5GS Interworking Suppression Control 335**
5GSIWKI suppression control 335
How 5GSIWKI suppression works 335
Configure 5GSIWKI suppression 336
Monitoring Subscriber Information 337

CHAPTER 33 **Intra-MME and Inter-MME Handover Procedures 341**
Feature Summary and Revision History 341
Summary Data 341
Revision History 341
Feature Description 341
How it Works 342
Call Flows 342
Inter-MME Handover Active-Active Transition Call Flow 342
Intra-MME Handover Active-Active Transition Call Flow 343
Inter/Intra-MME Handover Idle-Idle Transition Call Flow 344
Inter/Intra-MME Handover Active-Idle Transition Call Flow 345
Inter-MME Handover and Multi-PDN Handling Active-Idle Transition with ULI Change Call Flow 346
Inter-MME Handover with Bearer Context Marked for Removal Call Flow 347
Intra-MME and Inter-MME Handover Procedures OAM Support 349
Bulk Statistics 349

CHAPTER 34 **MCC/MNC Configuration in the SGW Service 351**

Feature Summary and Revision History 351
Summary Data 351
Revision History 351
Feature Description 351
How it Works 352
Call Flows 352
PLMN-type Detection Call Flow 352
Configuring the MCC or the MNC in the SGW Service 353
Configuration Example 353
OAM Support 354
Bulk Statistics Support 354

CHAPTER 35 **Message Interactions Support** 357
Feature Summary and Revision History 357
Summary Data 357
Revision History 357
Feature Description 358
How it Works 359
Call Flows 359
CBR Multi-PDN Call Flow 359
Graceful Stop the Existing PDN Procedure Call Flow 362
Inter MME Handover with Multi-PDN Handling (With PGW Interaction) Call Flow 365
Multi PDN Call X2 Handover SGW Relocation to cnSGW-C Call Flow 366
Multi-PDN S1 Handover SGW Relocation to Service-Pod (SGW) Call Fow 368
Multiple CBR for Same PDN Call Flow 371
Collision Resolver Discard Handling Call Flow 374
Suspend Handling Call Flow 375
Abort Handling of Low-Priority Procedure Call Flow 378
Double Delete Optimization Call Flow 379

CHAPTER 36 **Modify and Delete Bearer Command Support** 383
Feature Summary and Revision History 383
Summary Data 383
Revision History 383

Feature Description 383
How it Works 384
Call Flows 384
MBC Failure Handling Call Flow 384
MBC Success Handling Call Flow 385
DBC Failure Handling Call Flow 387
DBC Success Handling Call Flow 388

CHAPTER 37 **Modify Bearer Request Support** 391
Feature Summary and Revision History 391
Summary Data 391
Revision History 391
Feature Description 391
How it Works 392
Call Flows 392
UE-Triggered Service Request without PGW Interaction Call Flow 392
UE-Triggered Service Request with PGW Interaction Call Flow 393

CHAPTER 38 **Monitor Subscriber and Protocol Support** 397
Feature Summary and Revision History 397
Summary Data 397
Revision History 397
Feature Description 397
Feature Configuration 398
Configuring the Monitor Subscriber 398
Configuration Example 399
Configuring the Monitor Protocol 421
Configuration Example 422
Configuring the Transaction Messages 433
Configuration Example 433
Accessing the Logs 437

CHAPTER 39 **Multiple PDN Attach or Detach Procedures** 439
Feature Summary and Revision History 439

Summary Data 439
Revision History 439
Feature Description 439
How it Works 440
Call Flows 440
UE-requested PDN Connection Call Flow 440
UE-requested or the MME-requested PDN Disconnection Call Flow 443
PGW-requested Disconnection Call Flow 445

CHAPTER 40 **Performance Optimization Support** 449
Feature Summary and Revision History 450
Summary Data 450
Revision History 452
Feature Description 453
Async BG-IPC from GTPC-EP towards SGW-Service 453
Feature Description 453
Batch ID Allocation, Release, and Reconciliation Support 453
Feature Description 453
How it Works 453
Feature Configuration 454
OAM Support 455
Bulk Statistics 455
Cache Pod Optimization 456
Feature Description 456
CDL Flush Interval and Session Expiration Tuning Configuration 456
Feature Description 456
Feature Configuration 456
Configuration Example 457
DDN Call Flow Optimization 457
Feature Description 457
How it Works 457
Call Flows 457
Feature Configuration 461
Configuration Example 462

OAM Support 462
Bulk Statistics 462
DDN Timeout Configuration 462
Feature Description 462
Feature Configuration 462
Domain-based User Authorization Using Ops Center 463
Feature Description 463
How it Works 463
Feature Configuration 464
Configuration Example 465
Configuration Verification 465
Edge Echo Implementation 465
Feature Description 465
How it Works 466
OAM Support 466
Bulk Statistics Support 466
ETCD Peer Optimization Support 467
Feature Description 467
How it Works 467
Optimized GTPv2 Encoder and Decoder 468
Feature Description 468
Feature Configuration 469
Configuration Example 469
OAM Support 470
Bulk Statistics Support 470
GTPC Endpoint with GR Split 470
Feature Description 470
How it Works 470
GTPC Endpoint Interface Split with S11 and S5 471
Feature Description 471
How it Works 471
Feature Configuration 472
Configuration Example 473
GTPC IPC Cross-rack Support 473

Feature Description 473
How it Works 475
Call Flows 476
Feature Configuration 479
Configuration Example 480
OAM Support 480
KPI Support 481
Interservice Pod Communication 481
Feature Description 481
How it Works 482
Call Flows 482
OAM Support 484
Statistics Support 484
MBR Call Flow Optimization 484
Feature Description 484
How it Works 484
Call Flows 485
Feature Configuration 491
Configuration Example 492
Configuration Verification 492
OAM Support 492
Bulk Statistics Support 492
Maintenance Mode 493
Feature Description 493
How it Works 493
Limitations 494
Enabling or Disabling Maintenance Mode 494
Enabling or Disabling Maintenance Mode Example 494
Verifying the Maintenance Mode State 494
Partial CDL Update for Idle-Active Call Flow 495
Feature Description 495
How it Works 495
Limitations 496
Feature Configuration 496

Configuration Example 496
OAM Support 497
Bulk Statistics Support 497
PFCP Session Report with DLDR Throttling Support 497
Feature Description 497
How it Works 497
Feature Configuration 498
Configuration Example 498
Configuration Verification 499
OAM Support 499
Bulk Statistics Support 499
Throttling Support for Create Session Requests on S11 Interface 500
Feature Description 500
How it Works 500
Enable Throttling for Create Session Requests 500
Configuration Example 502
Configuration Verification 502
OAM Support 502
Bulk Statistics Support 502
Resiliency Handling 503
Feature Description 503
How it Works 503
Feature Configuration 504
Configuration Example 505
Configuration Verification 506
OAM Support 506
Bulk Statistics Support 506
Roaming Peer Path Management Optimization 506
Feature Description 506
How it Works 506
Feature Configuration 507
Configuring the Operator Policy and Subscriber Policy 507
Configuration Example 508
Configuring the Default Gateway 509

Configuration Example 509
Configuration Verification 510
Enable the echo request optimization for roaming peer node 510
OAM Support 514
Bulk Statistics Support 514
Flag DB Database Updates 514
Feature Description 514
OAM Support 515
Bulk Statistics Support 515
UDP Proxy Functionality Merged into Protocol Microservices 516
Feature Description 516
PFCP Protocol Endpoint with UDP Proxy Bypass 516
GTPC Protocol Endpoint with UDP Proxy Bypass 516

CHAPTER 41 **Presence Reporting Area** 521
Feature Summary and Revision History 521
Summary Data 521
Revision History 521
Feature Description 521
How it Works 522

CHAPTER 42 **NB-IoT and LTE-M RAT Type Support** 525
Enable RAT Type NB-IoT and LTE-M for CDR 526
Troubleshooting Information for RAT-Type Support 526

CHAPTER 43 **Redundancy Support** 529
Feature Summary and Revision History 529
Summary Data 529
Revision History 529
High Availability Support 530
Feature Description 530
High Availability of UDP Proxy 530
Architecture 530
cnSGW-C Pod and VM Deployment Layout 530

How it Works 531
Configuring Pod-level Labelling and Replicas 532
Configuration Example 532
Configuration Verification 532
Inter-Rack Redundancy Support 533
Feature Description 533
How It Works 533
Overview 533
Inter-Rack Redundancy Triggers 534
Rack NF Roles 535
General Guidelines 536
Instance Awareness 537
Configuring Inter-Rack Redundancy Instance 537
Configuring Endpoint Instance Awareness 538
Configuring Profile cnSGW-C5GSA GW Instance Awareness 539
Configuring cnSGW-C Endpoint 539
Dynamic Routing 541
Configuring Dynamic Routing Using BGP 544
Configuring BGP Speaker 547
IPAM 548
Configuring IPAM 549
Geo Replication 550
Configuring ETCD/Cachepod Replication 551
Geo Monitoring 552
Pod monitoring 552
Remote Cluster Monitoring 552
Traffic monitoring 553
BFD Monitoring 553
CDL GR Deployment 556
Prerequisites for CDL GR 556
CDL Instance Awareness and Replication 556
Lawful Intercept 560
RADIUS Configuration 561
Software Upgrade on GR Pairs 563

GR CLI 566
Geo Switch Role 566
Geo Reset Role 567
Troubleshooting 567
show/clear Commands 567
Monitor Subscriber 578
Monitor Protocol 579
Geographic Redundancy OAM Support 579
Prerequisites for RMA Process 580
Health Check 580
Recovery Procedure 585
Key Performance Indicators (KPIs) 586
Bulk Statistics 592
Alerts 595
Maintenance Mode 601

CHAPTER 44 **Service Configuration Enhancements** 603
Feature Summary and Revision History 603
Summary Data 603
Revision History 603
Feature Description 603
Feature Configuration 604
Configuring the SGW Profile 604
Configuration Example 604
Configuration Verification 604
Configuring the Subscriber Policy 605
Configuration Example 606
Configuring the Operator Policy 606
Configuration Example 606
Configuring the Policy DNN 606
Configuration Example 607
Configuration Modification Impact 608
Troubleshooting Information 609
Configuration Errors 609

CHAPTER 45 SGW Charging Support 611

Feature Summary and Revision History 611

Summary Data 611

Revision History 611

Feature Description 612

Architecture 613

Roaming Support 614

How it Works 614

Call Flows 614

URR Installation on Initial Attach Call Flow 614

SGW CDR Call Flow 616

URR Removal and CDR Reporting on Detach Call Flow 618

Usage Report on Hitting Threshold Call Flow 620

URR Installation for Dedicated Bearer Call Flow 622

URR Removal and CDR Generation on Deletion of Dedicated Bearer Call Flow 623

Volume Reporting on S11 Trigger Call Flow 625

Volume Reporting on S5 Trigger Call Flow 627

Standards Compliance 629

Limitations 629

Feature Configuration 630

CLI Configuration 630

Configuring the cnSGW-C Charging Profile or GTP Prime 631

Configuring the Charging Mode 636

Configuring the cnSGW-C Charging Threshold 636

Configuring cnSGW-C Charging Threshold and cnSGW-C Charging Profile Association 638

Configuring Call Control Profile 639

Configuring Charging Characteristics Under Call Control Profile 640

Show CLI 641

GTPP-EP SFTP Push CLI 641

CDR Fields Supported in cnSGW-CDRs 641

custom24 Dictionary 641

ASN.1 Definition for Fields in custom24 648

SGW Charging OAM Support 656

Bulk Statistics 656

CHAPTER 46 SGW Relocation Support 659

Feature Summary and Revision History 659

Summary Data 659

Revision History 659

Feature Description 659

How it Works 660

Call Flows 660

X2 Handover SGW Relocation to cnSGW-C Call Flow 660

S1 Handover SGW Relocation to cnSGW-C Call Flow 662

TAU X2 Handover SGW Relocation to cnSGW-C Call Flow 663

X2 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow 665

S1 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow 667

X2 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow 669

S1 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow 672

Inter and Intra MME Handover and S1 SGW Relocation with Less Number of Bearer Context Call Flow 675

SGW Relocation OAM Support 676

CHAPTER 47 Sx Load/Overload Control Handling 677

Feature Summary and Revision History 677

Summary Data 677

Revision History 677

Feature Description 678

How it Works 678

Node Feature Support 678

UP Selection 678

Handling Sx Messages during Peer Overload Scenario 679

Throttling Support for Sx Establishment 679

Session Termination Trigger From User-Plane in Self-Protection 679

Failure-handling Profile Support for Congestion Cause 679

Configuring the Sx Load/Overload Feature 680

Configuring Failure Handling Profile 681

Throttling and Exclusions of Sx Request Messages in Self-Overload State 682
Sx Load/Overload Control OAM Support 683
Bulk Statistics 683

CHAPTER 48 **Stale Session Handling and Clearing** 685
Feature Summary and Revision History 685
Summary Data 685
Revision History 685
Feature Description 686
How it Works 686
Call Flows 686
Timer Expiry Handling Call Flow 686
Feature Configuration 688
Configuration Example 689
Configuration Verification 689
OAM Support 689
Bulk Statistics 689

CHAPTER 49 **Support for CSFB Procedures Suspend and Resume** 691
Feature Summary and Revision History 691
Summary Data 691
Revision History 691
Feature Description 691
How it Works 692
Call Flows 692
Suspend Notification Call Flow 692
Resume Notification Call Flow 694
Empty Modify Bearer Request for Resume Call Flow 695

CHAPTER 50 **Update Bearer Request and Response** 699
Feature Summary and Revision History 699
Summary Data 699
Revision History 699
Feature Description 699

Standards Compliance 700
How it Works 700
Call Flows 700

CHAPTER 51 UPF Selection Support 705
Feature Summary and Revision History 705
Summary Data 705
Revision History 706
Feature Description 706
UPF Selection using DNN and DCNR Support 706
Feature Description 706
How it Works 706
UPF Selection Methods 707
Configuring UPF Selection Methods 708
Configuring UPF Group Profile-based UPF Selection 708
Configuring Network-based UPF Selection 708
Configuring Policy based UPF Selection 709
Troubleshooting Information 710
Configuration Errors 710
UPF Selection using Location Support 710
Feature Description 710
Configuring the UPF Selection Feature 711
Configuring ECGI for EPS 711
Configuring TAI-Group 712
Configuring Location-area-group 713
Configuring UPF Group and UPF Selection Policy Enhancement 713
SGW-U selection based on system diagnostics 715
Configure SUPI for SGW-U selection 715
Combined UPF Selection for cnSGW-C and SMF 716
Feature Description 716
Standards Compliance 716
How it Works 716
System Architecture 717
Call Flows 718

Configuring the Combined UPF Selection for cnSGW-C and SMF 725
Configuring Converged-Core Profile 725
Configuring Node-ID 726
Block UPF to handle continuous Sxa session creation failures 728
How handling continuous Sxa session creation failures works 729
Configure cnSGWc to handle continuous Sxa session creation failures 729
Enable UPF blocking 729
Reactivate blocked UPF 730
Monitoring and troubleshooting 731
Show command and output 732
Bulkstats 732
UPF Selection OAM Support 733
Bulk Statistics 733

CHAPTER 52 VoLTE Call Prioritization 735
Feature Summary and Revision History 735
Summary Data 735
Revision History 735
Feature Description 736
How it Works 736
Feature Configuration 736
Configuring the Priority 736
Configuration Example 737
Configuration Verification 737
Sx Message Priority 739
OAM Support 739
Bulk Statistics 739

CHAPTER 53 cnSGW-C Troubleshooting 741
Description 741
Using CLI Data 741
show subscriber and cdl show Commands 742
Logs 748
Logs for Event Failures 749

How it Works 751

Enable Event Logging 753

CHAPTER 54 **Sample cnSGW-C Configuration** 755

Sample Configuration 755



About this Guide

Note The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. While any existing biased terms are being substituted, exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

This preface describes the *Ultra Cloud Core Serving Gateway Control Plane Function - Configuration and Administration Guide*, the document conventions, and the customer support details.

- Conventions Used, on page xxxv
- Contacting Customer Support, on page xxxvi

Conventions Used

The following tables describe the conventions used throughout this documentation.

Notice Type	Description
Information Note	Provides information about important features or instructions.
Caution	Alerts you of potential damage to a program, device, or system.
Warning	Alerts you of potential personal injury or fatality. May also alert you of potential electrical hazards.

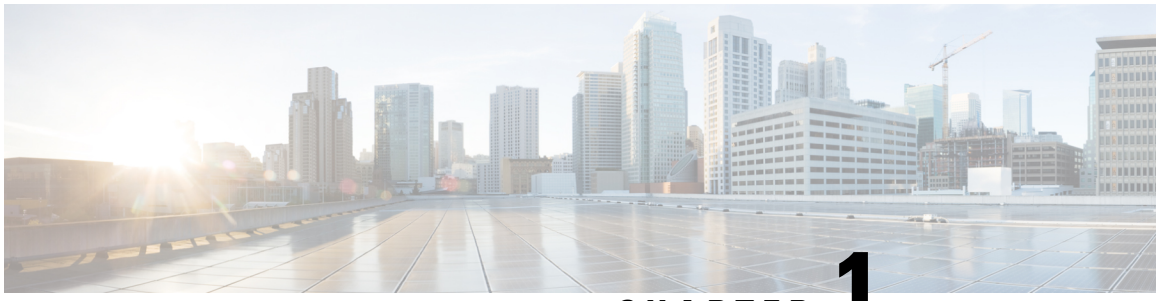
Typeface Conventions	Description
Text represented as a `screen display`	This typeface represents displays that appear on your terminal screen, for example: `Login:`

Typeface Conventions	Description
Text represented as **commands**	This typeface represents commands that you enter, for example: **show ip access-list** This document always gives the full form of a command in lowercase letters. Commands are not case sensitive.
Text represented as a **command** *variable*	This typeface represents a variable that is part of a command, for example: **show card** *slot_number* *slot_number* is a variable representing the applicable chassis slot number.
Text represented as menu or sub-menu names	This typeface represents menus and sub-menus that you access within a software application, for example: Click the **File** menu, then click **New**

Contacting Customer Support

Use the information in this section to contact customer support.

Refer to the support area of http://www.cisco.com for up-to-date product documentation or to submit a service request. A valid username and password are required to access this site. Please contact your Cisco sales or service representative for additional information.



CHAPTER 1

5G Architecture

- Architectural Overview, on page 1
- Subscriber Microservices Infrastructure Architecture, on page 2
- Control Plane Network Function Architecture, on page 4

Architectural Overview

The Ultra Cloud Core is Cisco's solution supporting 3GPP's standards for 5G new radio (NR) standalone (SA) mode. These standards define various network functions (NFs) based on the separation of control plane (CP) and user plane (UP), such as CUPS. This separation enhances network performance and capabilities.

Control Plane Network Functions

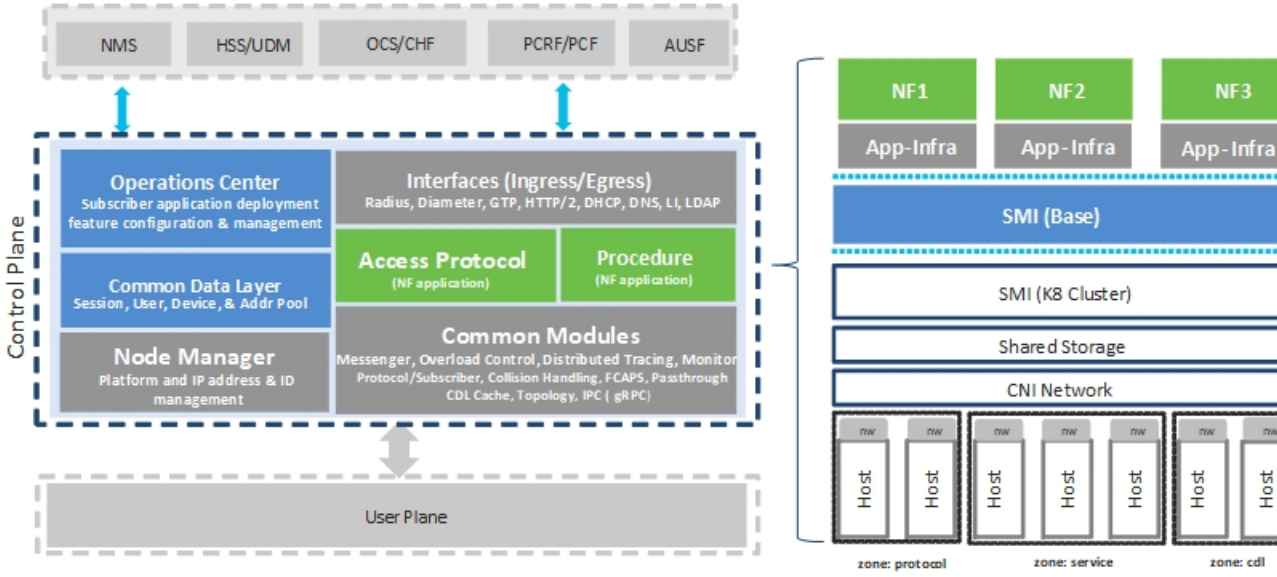
The Ultra Cloud Core's CP-related NFs use a common architecture with the following tenets:

- Cloud-scale—Fully virtualized for simplicity, speed, and flexibility.
- Automation and orchestration—Optimized operations, service creation, and infrastructure.
- Security—Multiple layers of security across the deployment stack from the infrastructure through the NF applications.
- API exposure—Open and extensive for greater visibility, control, and service enablement.
- Access agnostic—Support for heterogeneous network types (for example 5G, 4G, 3G, Wi-Fi, and so on).

Control plane NFs are designed as containerized applications, such as microservices. They deploy through the Subscriber Microservices Infrastructure (SMI).

The SMI defines the common application layers for functional aspects of the NF such as life-cycle management (LCM), operations and management (OAM), and packaging.

Figure 1: Ultra Cloud Core CP Architectural Components



User Plane Network Function

The 5G UP NF within the Ultra Cloud Core is the User Plane Function (UPF). Unlike the CP-related NFs, The 5G UPF uses the same Vector Packet Processing (VPP) technology as the Cisco 4G CUPS architecture. This commonality delivers a consistent set of capabilities between 4G and 5G, such as:

- Ultrafast packet forwarding.
- Extensive integrated IP Services such as Subscriber Firewall, Tethering, Deep-Packet Inspection (DPI), Internet Content Adaption Protocol (ICAP), Application Detection and Control (ADC), and header enrichment (HE).
- Integrated third-party applications for traffic and TCP optimization.

Subscriber Microservices Infrastructure Architecture

The Ultra Cloud Core (UCC) Subscriber Microservices Infrastructure (SMI) is a layered stack of cloud technologies. It enables rapid deployment and seamless life-cycle operations for microservices-based applications:

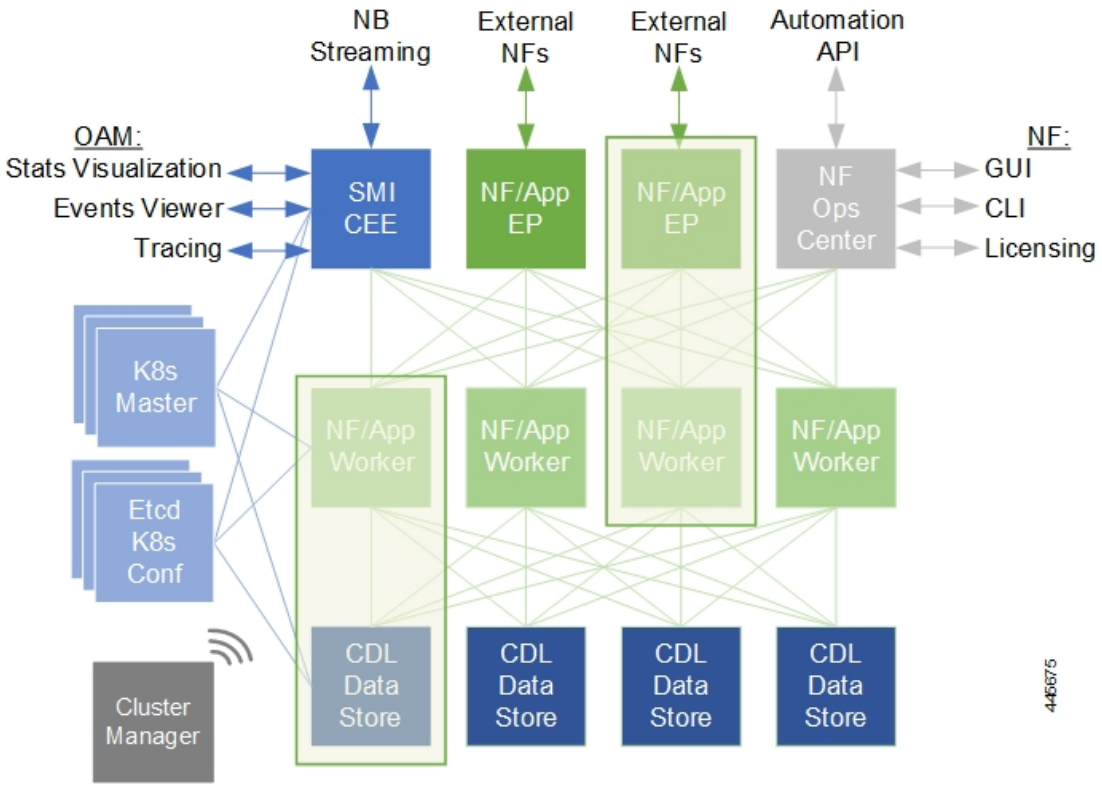
The SMI stack consists of the following components:

- SMI Cluster Manager—Creates the Kubernetes (K8s) cluster, creates the software repository, and provides ongoing LCM for the cluster, including deployment, upgrades, and expansion.

- Kubernetes Management—Includes the K8s primary and etcd functions, which provide LCM for the NF applications that are deployed in the cluster. This component also provides cluster health monitoring and resources scheduling.
- Common Execution Environment (CEE)—Offers utilities and OAM functionalities for Cisco Cloud native NFs and applications. These include licensing and entitlement functions, configuration management, and telemetry. It also provides alarm visualization, logging management, and troubleshooting utilities. Also, it provides consistent interaction and experience for all customer touch points and integration points in relation to these tools and deployed applications.
- Common Data Layer (CDL)—Provides a high performance, low latency, stateful data store, designed specifically for 5G and subscriber applications. This next generation data store offers high availability in local or geo-redundant deployments.
- Service Mesh—Provides sophisticated message routing between application containers, enabling managed interconnectivity, extra security, and the ability to deploy new code and new configurations in low risk manner.
- NB Streaming—Provides Northbound Data Streaming service for billing and charging systems.
- NF or Application Worker Nodes—The containers that comprise an NF application pod.
- NF or Application Endpoints (EPs)—The NFs or applications and their interfaces to other entities on the network
- Application Programming Interfaces (APIs)—Provides various APIs for deployment, configuration, and management automation.

The following figure depicts how these components interconnect to comprise a microservice-based NF or application.

Figure 2: SMI Components



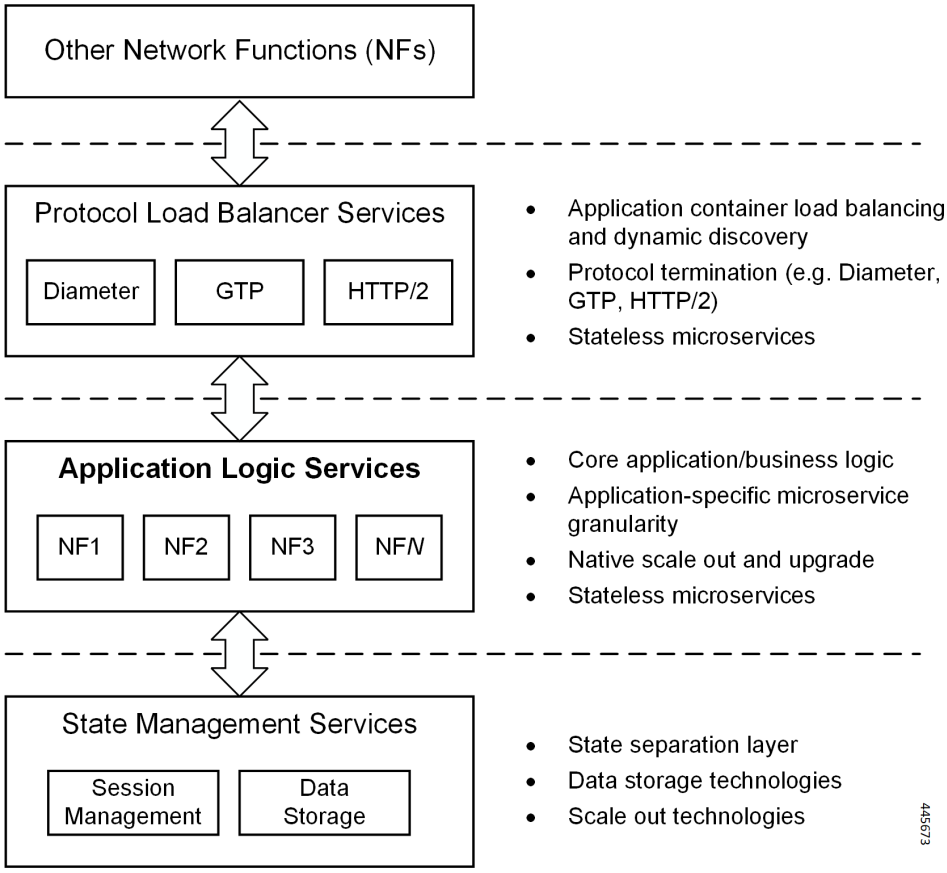
For more information on SMI components, see Ultra Cloud Core Subscriber Microservices Infrastructure and the related-documentation at *Deployment Guide* > *Overview* chapter.

Control Plane Network Function Architecture

Control plane (CP) NFs use a three-tiered architecture. This architecture leverages stateful or stateless capabilities in cloud native environments. The architectural tiers are as follows:

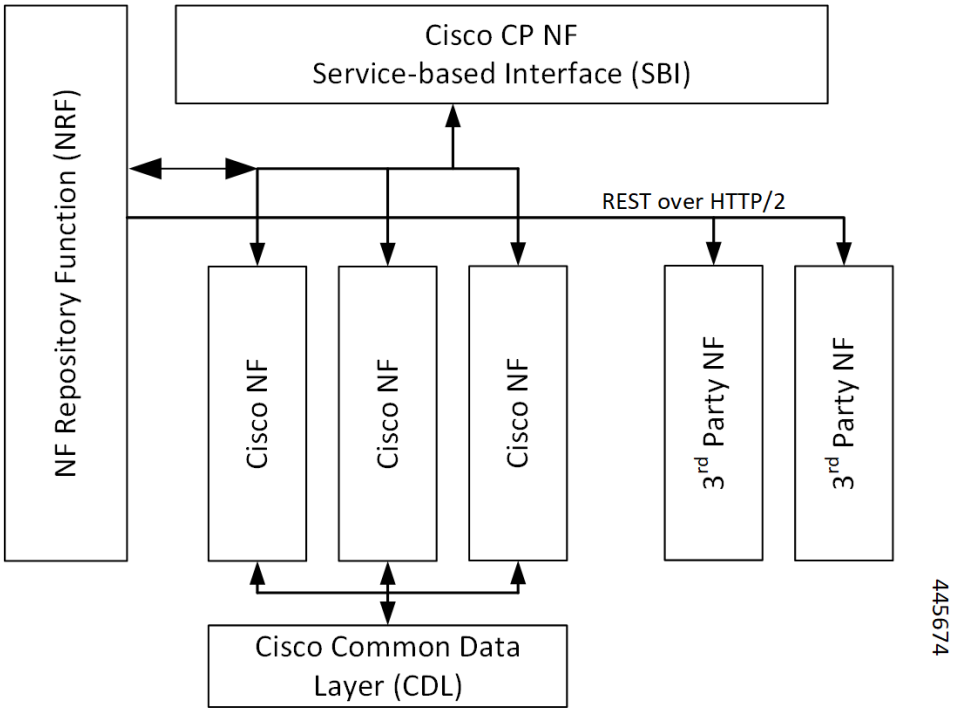
- Protocol Load Balancer Services—These are stateless microservices that are primarily responsible for dynamic discovery of application containers as well as for protocol proxy and termination. These include traditional 3GPP protocols and new protocols that are introduced with 5G.
- Applications Services—Responsible for implementing the core application or business logic, these are the stateless services that render the actual application based on the received information. This layer may contain varying degrees of microservice granularity. Application services are stateless.
- State management services—Enable stateless application services by providing a common data layer (CDL) to store or cache state information (for example session and subscriber data). This layer supports various data storage technologies from in-memory caches to full-fledge databases.

Figure 3: Control Plan Network Function Tiered Architecture

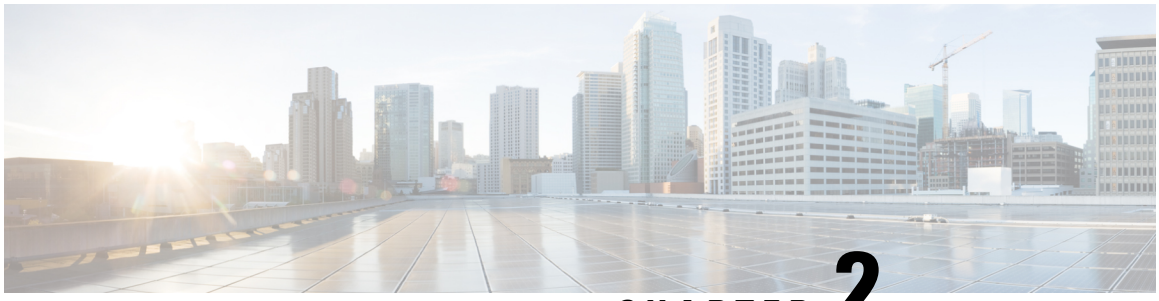


The three-tiered architecture on which Cisco CP NFs are designed fully support the 5G core (5GC) Service-based Architecture (SBA) defined by 3GPP. These NFs communicate using the Service-based Interface (SBI). They use HTTP/2 over TCP, as defined by 3GPP.

Figure 4: Cisco CP NF Service-based Architecture Support



For more information on the Cisco network functions, see their corresponding network function documentation.



CHAPTER 2

cnSGW-C Overview

- Product Description, on page 7
- Converged Core Overview, on page 7
- Converged core architecture and ULB integration, on page 8
- ULB integration in converged core, on page 9
- ULB integration design, on page 9
- ULB integration of GTPC ingress/egress traffic, on page 10
- ULB integration of PFCP ingress/egress traffic, on page 11
- Use Cases, on page 12
- Deployment Architecture and Interfaces, on page 15
- Life Cycle of Data Packet, on page 20
- License Information, on page 20
- Standards Compliance, on page 20

Product Description

cnSGW-C is a Control Plane Network Function (NF) of the converged core network (4G-5GC). The Serving Gateway Control Plane Function (cnSGW-C) is built on top of the SMI architecture. cnSGW-C acts as a UE anchor and supports mobility procedures, along with session setup and termination procedures, as specified in *3GPP TS 23.401* and *3GPP TS 23.214*.

The Serving Gateway Control Plane Function (cnSGW-C) provides the functionality of the S-GW as defined by *TS 23.401 [2]*, except for the functions that are performed by the SGW-U, as described in *3GPP Spec 23.214 Table 4.3.2-1*. In addition, the cnSGW-C is responsible for selecting the SGW-U (as described in 3*GPP Spec 23.214 clause 4.3.3*) and for controlling the SGW-U with respect to the functions described in *TS 23.214 Table 4.3.2-1*.

With SMF (IWF) support based on Cisco Cloud Native Platform, it is recommended to support cnSGW-C functionality on Cloud Native Platform for better hardware utilization and O&M activities.

Converged Core Overview

The converged core solution provides an advanced, cloud-native, converged control plane with the capability to support 4G and 5G devices, and use cases.

Important This release supports only the cloud-native integrated S-GW and 5GSA GW instance with S5C and cnSGW-C functionalities.

The converged core solution removes the operational complexity by providing a unified core network to handle all types of subscribers and use cases.

The operator has the following benefits:

- Improves the overall network efficiency by reducing signaling between cnSGW-C and 5GSA GW while handling a 4G subscriber or handoff from 5G to 4G coverage area.
- Reduces latency introduced due to the extra hop SGW-U for a subscriber in 4G coverage area, by collapsing the data path in the Converged UPF, thus improving the overall user experience.
- Provides ability to use a unified subscriber policy and billing infrastructure using SBA interfaces for 4G and 5G devices.

The solution supports the following converged control plane and user plane functions:

- Converged Control Plane Functions
  - Integrates S-GW and 5GSA GW network functions as a single deployment, under a single Kubernetes namespace, to support 4G and 5G devices from E-UTRAN/NR (converged core gateway)
  - Supports logical network functions (data)
- Converged User Plane Functions
  - Integrates UPF and SGW-U functionalities as a single network function
  - Provides simultaneous support for N4 and Sxa interfaces
  - Terminates multiple control planes in a single deployment

Converged core architecture and ULB integration

The Converged Core architecture integrates the Unified Load Balancer (ULB) with the cnSGW-C Service Pods to support direct routing of GTPC and PFCP signaling traffic. By eliminating the need for dedicated GTPC and PFCP protocol pods for ingress and egress signaling, the architecture reduces signaling latency, simplifies signaling paths, and improves scalability in cloud-native deployments.

The ULB distributes ingress signaling messages to the appropriate cnSGW-C Service Pod based on the configured load-balancing policy and routes egress signaling messages to the corresponding peer network element. The ULB maintains session affinity to ensure that all signaling messages associated with an active subscriber session are consistently processed by the same Service Pod.

The ULB provides a common routing framework for GTPC and PFCP signaling, enabling direct communication between the ULB and the cnSGW-C Service Pods. This architecture supports efficient session processing, consistent message routing, and protocol-specific peer management for supported signaling procedures.

The Converged Core ULB integration provides the following capabilities:

- GTPC ingress and egress signaling message routing

- PFCP ingress and egress signaling message routing
- Session affinity for subscriber sessions
- GTP and PFCP peer management

ULB integration in converged core

The Unified Load Balancer (ULB) integrates GTPC and PFCP signaling into the Converged Core architecture by forwarding ingress and egress signaling traffic directly to Service Pods for processing. This architecture replaces the legacy deployment model in which GTPC and PFCP signaling was processed by dedicated protocol pods.

In a ULB-enabled deployment, GTPC and PFCP signaling is processed within the Service Pod together with subscriber session management. The ULB distributes signaling traffic across available Service Pods while maintaining session affinity throughout the lifetime of the subscriber session. All signaling messages associated with a subscriber session are routed to the same Service Pod unless traffic is redistributed because of a Service Pod or node failure.

Processing GTPC and PFCP signaling within the Service Pod reduces inter-pod signaling and simplifies the signaling path between peer network functions and subscriber session processing. The ULB also supports traffic redistribution during Service Pod and node failures to maintain service continuity.

ULB integration design

The ULB integration uses the existing Unified Load Balancer infrastructure to distribute protocol signaling traffic across the available cnSGW-C Service Pods. The ULB selects a Service Pod for new session establishment and maintains routing information for subsequent protocol messages associated with the session.

For new sessions, the ULB distributes the initial signaling request across the available Service Pods. The selected Service Pod establishes ownership of the subscriber session and maintains the session information for subsequent signaling procedures.

For existing sessions, the ULB identifies the Service Pod associated with the session and forwards all subsequent signaling messages to the same Service Pod, ensuring session affinity throughout the session lifecycle.

After processing the signaling request, the Service Pod generates the appropriate response and sends it through the ULB to the peer network element.

This routing mechanism is used for both GTPC and PFCP signaling traffic.

The ULB integration feature is supported in the following deployment models:

- Flex
- Flex-small

Before directing signaling traffic through the ULB, deploy the required number of Service Pods and ensure that the ULB is operational. During normal operation, the ULB maintains session affinity throughout the subscriber session lifecycle. If a Service Pod or node becomes unavailable, the ULB redistributes signaling traffic to an available Service Pod to maintain service continuity.

Session routing and affinity

The ULB maintains session affinity to ensure that all protocol messages belonging to the same subscriber session are forwarded to the same cnSGW-C Service Pod.

For a new subscriber session, the ULB selects a Service Pod based on the configured load-balancing algorithm. After the session is established, the routing information is maintained so that subsequent protocol messages associated with the session continue to be processed by the selected Service Pod.

Session affinity applies to both GTPC and PFCP session-level signaling and remains valid throughout the lifetime of the subscriber session.

Call flow summary

The ULB integration supports the complete lifecycle of GTPC and PFCP signaling procedures, including session establishment, session modification, session deletion, and session release. During message processing, the ULB forwards signaling traffic to the appropriate Service Pod, where subscriber session management and protocol processing are performed. The Service Pod returns the response through the ULB to the corresponding peer network element.

ULB integration of GTPC ingress/egress traffic

Table 1: Feature history

Feature name	Release	Feature description
ULB integration of GTPC traffic	2026.03.0	This release introduces ULB-based ingress and egress routing for GTPC traffic, enabling scalable traffic distribution and session-aware message steering. The enhancement optimizes control-plane traffic handling and supports efficient scaling in cloud-native deployments.

The ULB routes GTPC ingress and egress signaling messages to the appropriate cnSGW-C Service Pod while maintaining session affinity. GTPC signaling is processed directly by the Service Pod without requiring dedicated GTPC protocol pods for ingress and egress signaling.

GTPC message routing

The ULB receives GTPC ingress signaling messages from peer network elements and forwards them to the appropriate Service Pod based on the configured load-balancing policy.

For a new subscriber session, the ULB distributes the initial GTPC signaling request to a Service Pod. The selected Service Pod processes the request and establishes the routing information required for subsequent GTPC messages.

After the session is established, the ULB maintains session affinity and forwards all subsequent session-level GTPC messages to the same Service Pod. Egress GTPC messages generated by the Service Pod are routed through the ULB to the appropriate peer network element.

The ULB supports routing of both ingress and egress GTPC signaling while maintaining consistent session processing.

GTP peer management

The Unified Load Balancer supports GTP peer management by forwarding GTP peer management messages to the appropriate processing component while maintaining independent routing for session-level signaling. Session-level GTPC signaling continues to be routed based on session affinity to the owning Service Pod, ensuring consistent session processing.

Limitations

Automatic redistribution of GTPC signaling traffic during Service Pod or node failures is not supported in the current release. When a Service Pod or node becomes unavailable, the ULB does not automatically update its forwarding information to reroute subsequent signaling messages to another Service Pod. Manual intervention may be required to restore normal signaling traffic.

ULB integration of PFCP ingress/egress traffic

Table 2: Feature history

Feature name	Release	Feature description
ULB integration for PFCP traffic	2026.03.0	This release introduces ULB-based ingress and egress routing forPFCP traffic, enabling scalable traffic distribution. The enhancementoptimizes PFCP control-plane traffic handling and supportsefficient scaling in cloud-native deployments.

The ULB routes PFCP ingress and egress signaling messages to the appropriate cnSGW-C Service Pod while maintaining session affinity. PFCP session-level signaling is processed directly by the Service Pod, while node-level peer management continues to use the existing PFCP processing framework.

PFCP message routing

The ULB receives PFCP ingress signaling messages from peer network elements and forwards them to the appropriate Service Pod based on the configured load-balancing policy.

For a new PFCP session or association, the ULB distributes the initial PFCP request to a Service Pod. The selected Service Pod processes the request and establishes the routing information required for subsequent PFCP messages.

After the session or association is established, the ULB maintains session affinity and forwards all subsequent session-level PFCP messages to the same Service Pod. Egress PFCP messages generated by the Service Pod are routed through the ULB to the appropriate peer network element.

The ULB supports routing of both ingress and egress PFCP signaling while maintaining consistent session processing.

PFCP peer management

The Unified Load Balancer differentiates between PFCP node-level and session-level signaling and routes messages to the appropriate processing component.

Node-level PFCP messages continue to use the existing PFCP processing framework, while session-level PFCP messages are forwarded to the Service Pod. The Service Pod processes PFCP session establishment, modification, and deletion procedures and maintains session affinity so that all messages belonging to an active PFCP session are consistently processed by the same Service Pod.

This routing mechanism enables separate handling of node-level and session-level PFCP signaling while maintaining consistent message processing and session affinity across the Converged Core deployment.

Use Cases

This section describes the use cases that cnSGW-C supports:

- **cnSGW-C Configuration**

  The cnSGW-C base configuration provides a detailed view of configurations required for the cnSGW-C to be operational. The configuration includes setting up the infrastructure to deploy the cnSGW-C, deploying the cnSGW-C through SMI, and configuring the Ops Center for exploiting the cnSGW-C capabilities over time. For more information on SMI, see the *Ultra Cloud Core SMI Cluster Deployer Operations Guide*.

  The following features are related to this use case:

  - APN Profile Support, on page 91
  - Service Configuration Enhancements, on page 603
  - UPF Selection Support, on page 705

  For Converged Core deployment, cnSGW-C is deployed using Converged Ops Center.

- **Session Management**

  Every UE accessing the EPC is associated with a single S-GW. cnSGW-C supports multiple PDN for given UE. As a part of Session Management, cnSGW-C supports the following:

  - Default and dedicated bearer establishment
  - Bearer modification
  - Bearer deactivation

  The following features are related to this use case:

  - 3GPP RAN/NAS Cause Codes Support, on page 79
  - Change Notification Request Handling, on page 101
  - Context Replacement Support, on page 113
  - Dedicated Bearer Support, on page 121
  - Delete Bearer and Delete Session Request, on page 133
  - DSCP Marking for CP Signaling Messages, on page 164

- eMPS and WPS Support, on page 199
- Emergency Call Support, on page 185
- Idle Session Timeout Settings, on page 313
- Initial Attach Support, on page 321
- Multiple PDN Attach or Detach Procedures, on page 439
- Presence Reporting Area, on page 521
- Update Bearer Request and Response, on page 699
- VoLTE Call Prioritization, on page 735

- **Support for UE Mobility**

  cnSGW-C is a mobility anchor point for UE. In LTE Network, there can be mobility between eNodeB to eNodeB, with or without MME change. UE can also move from one cnSGW-C to another cnSGW-C with different modes, S1-based Relocation, X2-based Relocation, and 5G-4G interworking.

  The following features are related to this use case:

  - IDFT Support, on page 297
  - Intra-MME and Inter-MME Handover Procedures, on page 341
  - Modify Bearer Request Support, on page 391
  - Presence Reporting Area, on page 521
  - SGW Relocation Support, on page 659

- **S1-Release/Buffering/Downlink Data Notification**

  cnSGW-C handles releasing S1-U bearer between eNodeB and SGW-U. When cnSGW-C receives Radio Access Bearers (RAB) message indicating that S1-U bearers are released, it updates User Plane and moves UE to IDLE state. When in IDLE state, if UE receives downlink data packet, cnSGW-C generates DDN message towards MME to page UE.

  cnSGW-C also supports DDN Throttling, DDN Delay, and High Priority feature for DDN.

  The following features are related to this use case:

  - Access Bearer Release Support, on page 87
  - Downlink Data Notification, on page 139
  - DDN Advance Features, on page 150

- **Retransmission and Timeout**

  For all procedures, as per *3GPP TS 23.401/29.274*, cnSGW-C supports N3-Retransmission, and T3-Timeout Support. These are supported for S11, S5, and Sx interfaces.

  The following feature is related to this use case:

  - GTPv2 and Sx Messages Retransmission and Timeout Handling, on page 273

- **Failure and Error Handling**

cnSGW-C supports handling of:

- Failure response for Create Session Request as part of initial attach procedure and additional PDN setup procedure
- PGW-initiated Dedicated Bearer Creation (DBC) procedure failure scenario
- Radio Access Bearers (RAB), Modify Bearer Request and Response (MBR) from PGW and User Plane

The following feature is related to this use case:

- Failure and Error Handling Support, on page 213

- **Load/overload Control Functions**

  cnSGW-C supports:

  - Exchange of load/overload control information and actions during peer node overload over Sx interface.
  - Handling load/overload information on GTPv2 interface.

  The following features are related to this use case:

  - GTPv2 Load/Overload Support, on page 279
  - Sx Load/Overload Control Handling, on page 677

- **cnSGW-C Charging Support**

  cnSGW-C supports:

  - Offline Charging (Gz).
  - Writing CDR to local disk storage. The CDR files are pushed to SFTP server periodically.
  - CDR generation for selected subscribers. This is achieved by enabling CDR generation per Operator Policy through call control profile.

  The following feature is related to this use case:

  - SGW Charging Support, on page 611

- **Peer and Path Management for GTPC and Sx**

  cnSGW-C supports:

  - Peer management for MME (S11 peers), PGW (S5 Peers), and User Plane.
  - Peer monitoring through ECHO Request/Response and Heartbeat Request/Response.
  - Handling of path failure events for S11 and S5 peers.

  The following features are related to this use case:

  - GTPC and Sx Path Management, on page 226
  - GTPC Path Failure, on page 232

- Customization of Path Failure Detection, on page 237
- Sx Path Failure, on page 235

- **Redundancy Support**

  The cnSGW-C deployment in K8 cluster plays a vital role to support High Availability (HA) and Geographic Redundancy (GR).

  The Redundancy Support ensures stateful session continuity among the clusters during the rack or cluster failures.

  The cnSGW-C achieves the HA through redundant set-up of each cluster component such that any single point of failure is avoided.

  The GR provides rack-level redundancy to replicate data between two separate K8 Clusters across rack. On RACK/Cluster failure, traffic switches to a remote RACK to process the traffic. The failure can be due to power failure, multi-compute failures, network failure, multi-POD failure, BFD link failure, and so on.

  The following features are related to this use case:

  - Redundancy Support, on page 529
  - High Availability Support, on page 530

- **Dynamic Routing**

  Dynamic routing enables L3 peering with Leafs, in addition to L2 Static routing.

  The following feature is related to this use case:

  - Dynamic Routing by Using BGP, on page 169

- **GTPU Path Management and Session Management**

  The UPF notifies an Error Indication message for a GTP-U peer to the sender when a GTP-PDU is received with a TEID that does not exist. This ensures that there are no stale sessions or bearers, and maintains consistency in the network.

  Error Indication and GTP-U Path Failure Indication communication between S-GW and UPF nodes is supported over the N4 interface. For the neighbor nodes, the communication is supported over the S1u/S5u interfaces. Behavior variations of local-purge or signal-peer for Error Indication and GTP-U Path Failure are considered in this implementation.

  The following features are related to this use case:

  - GTPU Error Indication, on page 243
  - GTPU Path Failure, on page 265

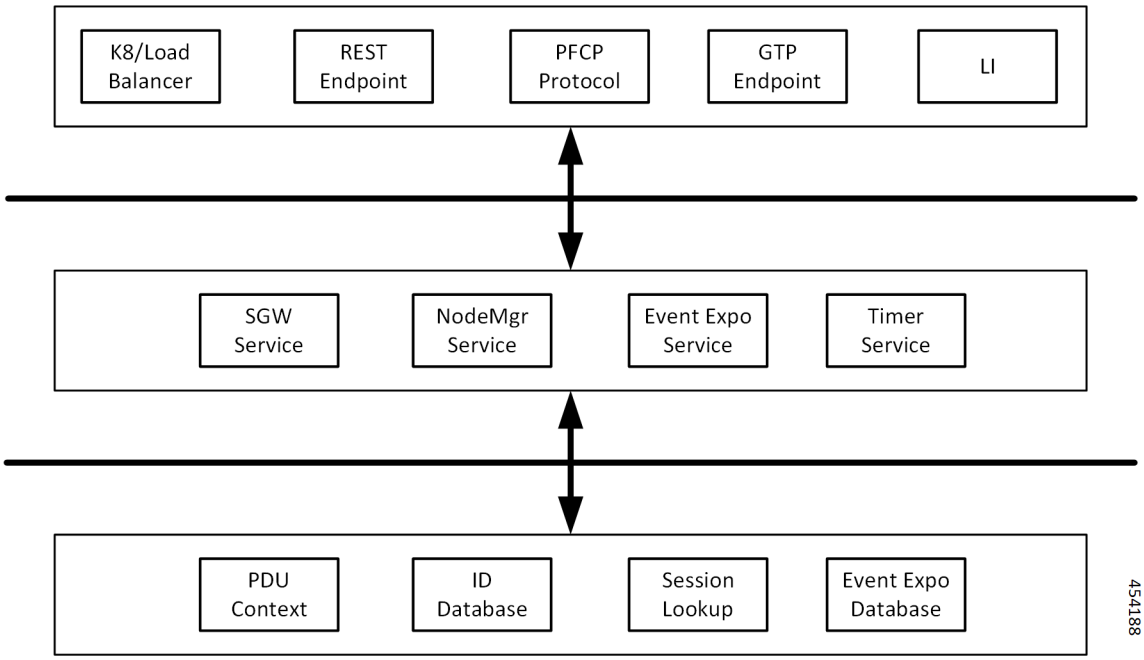
Deployment Architecture and Interfaces

cnSGW-C is a part of the converged core network functions portfolio with a common mobile core platform architecture. The core network functions include Access and Mobility Management Function (AMF), Policy Control Function (PCF), Session Management Function (SMF), and User Plane Function (UPF).

cnSGW-C Architecture

cnSGW-C network function consists of loosely coupled microservices. The microservice decomposition is based on a three-layered architecture, as illustrated in the following figure:

Figure 5: cnSGW-C Architecture



The following are the three layers of the cnSGW-C architecture:

- Layer 1 - Protocol and Load Balancer services (Stateless)
- Layer 2 - Application services (Stateless)
- Layer 3 - Database services (Stateful)

cnSGW-C Deployment

cnSGW-C NF is deployed in a separate namespace as an independent NF.

Figure 6: cnSGW-C Deployment

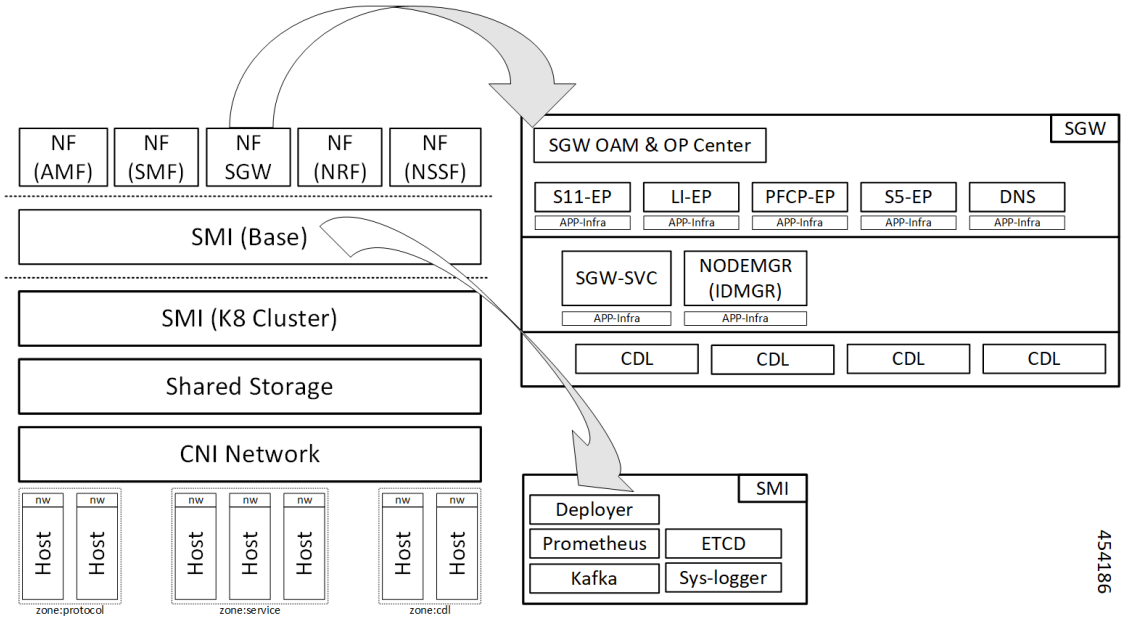
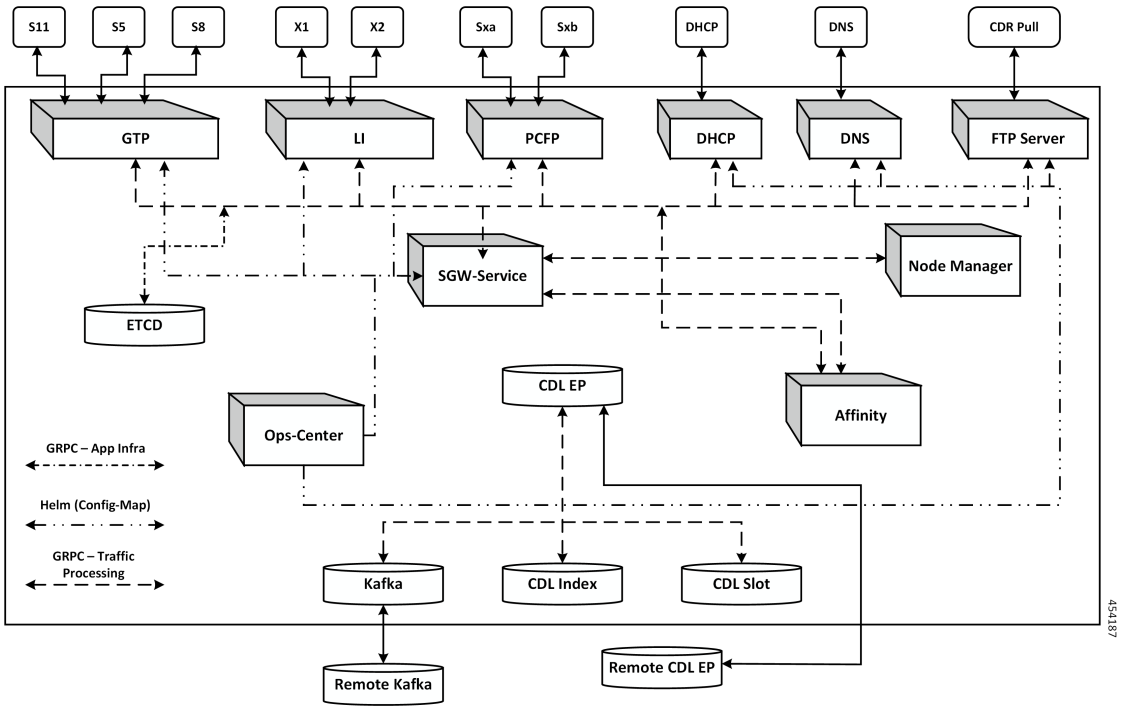


Figure 7: cnSGW-C HELM Chart



Converged Core Architecture

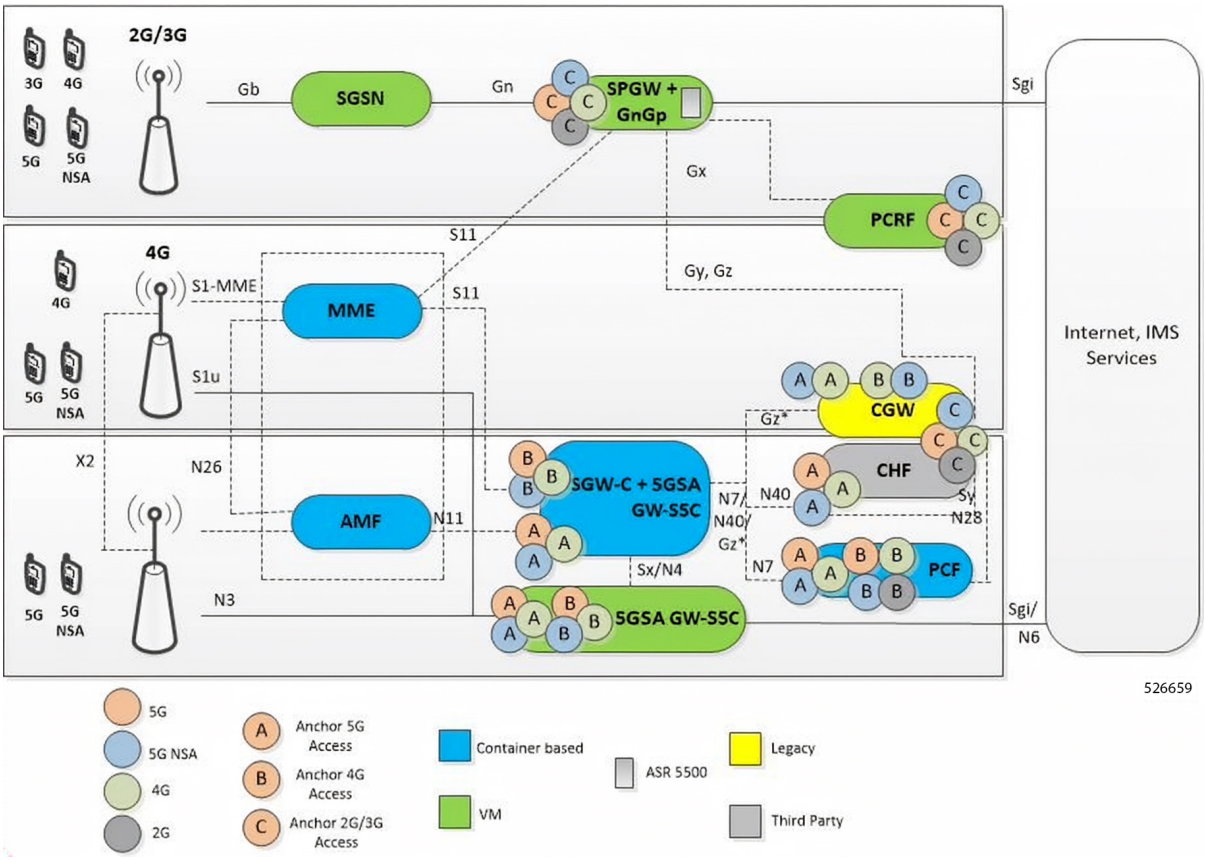
The converged core solution provides a single unified platform which is based on SMI architecture. The supporting architecture integrates the cloud-native S-GW and 5GSA GW deployment with 5GC and cnSGW-C functionalities. The solution uses 3GPP-defined SBA interfaces for policy and charging functions.

In the converged core architecture, the 4G and 5G capable UEs are anchored on the same control plane instance. The control plane instance provides the 5GSA GW, 5GC, and cnSGW-C functionalities.

The handoffs between 4G and 5G access types are seamless for 5G capable devices. The handoffs from LTE to UTRAN (bi-directional communication between 4G/5G and 3G/2G) are not seamless for 4G capable devices.

The following figure illustrates the supported network architecture.

Figure 8: Converged Core Architecture



The UPF deployed as a part of this solution is a VPC-SI VM. The UPF deployment is VM-based, and supports:

- SGW-U, PGW-U, and UPF functionalities in the same instance, and exposes the Sxa, Sxb, Sxab, or N4 interface towards the control plane.
- Multiple CP instances (up to 4) simultaneously.

Converged Core Deployment

The converged core deployment is based on the converged control plane and unified user plane infrastructure for all use cases.

In the converged core deployment, all 4G and 5G-capable UEs are anchored on the 5G core (5GSA GW) with SBA interfaces towards PCF.

The converged core deployment has a converged Ops Center that allows the configuration of cnSGW-C and 5GSA GW services along with other microservices. A single product helm chart is used to install components.

The following figure illustrates the Kubernetes deployment for the converged S-GW and 5GSA GW network function.

Figure 9: Kubernetes Deployment

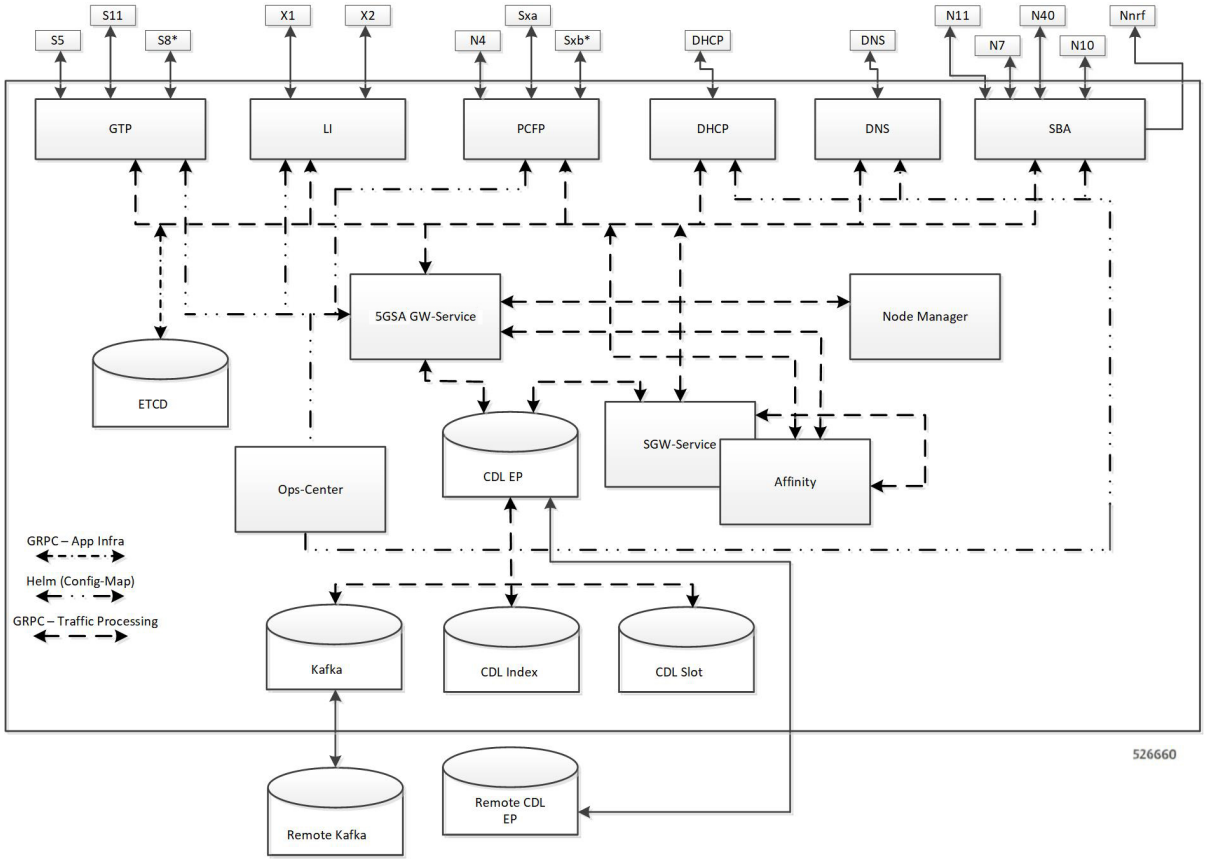


Figure 10: Kubernetes Deployment

The protocol layer services are shared across 5GSA GW and S-GW. The GTP endpoint terminates the S11 interface and S5/S8 interface. Similarly, the PFCP (protocol) endpoint terminates the N4 and Sxa interfaces.

The 5GSA GW and S-GW services are deployed as distinct pods and the session processing is segregated. Both the service pods use CDL for storing subscriber sessions.

Supported Interfaces

This section describes the interfaces supported between cnSGW-C and other network functions in the 5GC.

- S11—Reference point between the SGW and the MME
- S5/S8—Reference point between the SGW and the PGW/SMF
- Sxa—Reference point between the SGW-C and the SGW-U
- Gz—Reference point between the SGW-C and the Charging Server

Life Cycle of Data Packet

For information on life cycle of a data packet, see Initial Attach Support, on page 321.

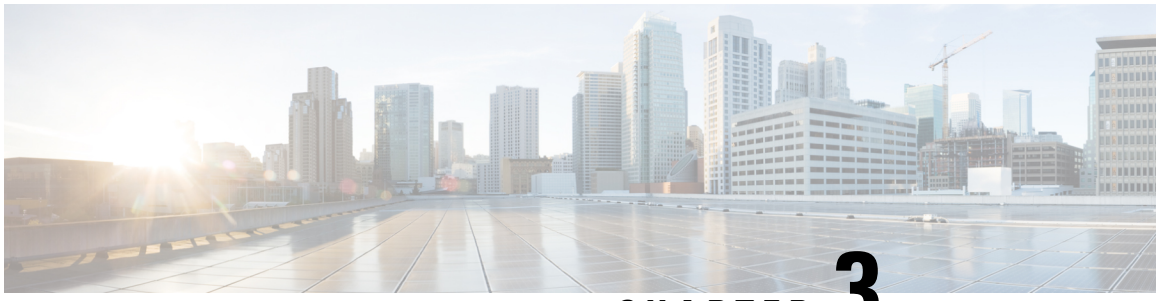
License Information

cnSGW-C supports Cisco Smart Licensing, a cloud-based licensing model that simplifies the purchase, deployment, and management of Cisco software assets. For more information, refer to the Smart Licensing Support collection page.

Standards Compliance

cnSGW-C complies with the following 3GPP standards:

- *3GPP TS 23.401 "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access"*
- *3GPP TS 23.402 "Architecture enhancements for non-3GPP accesses"*
- *3GPP TS 29.274 "Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C);"*
- *3GPP TS 23.214 "Architecture enhancements for control and user plane separation of EPC nodes"*
- *3GPP TS 29.244 "Interface between the Control Plane and the User Plane nodes"*
- *3GPP TS 24.008 "Mobile radio interface Layer 3 specification; Core network protocols; Stage 3"*
- *3GPP TS 23.007 "Restoration procedures"*
- *3GPP TS 22.153 "Multimedia priority service"*
- *3GPP TS 33.107 "3G security; Lawful interception architecture and functions"*



CHAPTER 3

Deploying and Configuring cnSGW-C through Operations Center

- Feature Summary and Revision History, on page 21
- Feature Description, on page 22
- cnSGW-C Service Configuration, on page 23
- Deploying and Accessing cnSGW-C, on page 24
- Loading Day 1 Configuration, on page 27

Feature Summary and Revision History

Summary Data

Table 3: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 4: Revision History

Revision Details	Release
The following enhancements were introduced: • Multiple entitlement tags • cnSGW-C deployment on bare metal server	2021.02.0

Revision Details	Release
First introduced.	2020.07.0

Feature Description

cnSGW-C deployment process involves deploying cnSGW-C through Subscriber Microservices Infrastructure (SMI) Cluster Deployer. You can perform configurations or customizations through the cnSGW-C Ops Center which is based on the Confd CLI.

cnSGW-C Ops Center

The Ops Center is a system-level infrastructure that provides the following user interface to:

- Trigger the deployment of microservices by providing variable helm chart parameters. These chart parameters control the scale and properties of Kubernetes objects (deployment, pod, services, and so on) associated with the deployment.
- Push application specific configuration to one or more micro-services through Kubernetes configuration maps.
- Issue application-specific execution commands (such as show commands and clear). These commands:
  - Invoke APIs in application-specific pods
  - Display the information returned by the application on the user interface

The following screenshot is a sample of the web-based CLI.

Figure 11: Web-based Ops Center



The cnSGW-C Ops Center allows you to configure the features, such as licensing, cnSGW-C engine, EGPT and PFCP endpoint, and CDL.

Prerequisites

Before deploying cnSGW-C on the SMI layer:

- Ensure that all the virtual network functions (VNFs) are deployed.
- Run the SMI synchronization operation for the cnSGW-C Ops Center and Cloud Native Common Execution Environment (CN-CEE).

cnSGW-C Service Configuration

The cnSGW-C service requires the basic configuration to process Call Setup, Modify, and Delete Request.

Mapping Pods with Node Labels

Prerequisites

- Ensure that the node labels are according to the pod deployment layout.
- Ensure that the external VIPs are according to the requirement of NF.
- Enable Istio for pod to pod traffic load balancing.

Node Labels are key and value pairs that are attached to nodes at cluster synchronization. Each node can have a set of key and value labels defined. Each key must be unique for a node. With labels, users can map their NF pods onto nodes in a loosely coupled manner.

Important

- The pod-level labeling configuration is applicable only when the cnSGW-C is deployed on a bare metal server.
- Ensure to configure the node label on the SMI cluster deployer before mapping the pods. Following is the sample command for master-1 labeling:

```
[cndp-clpnc-cm-cm-primary] SMI Cluster Deployer (config-nodes-master-1)# k8s node-labels smi.cisco.com/svc-type smf-node
```

To map the pods with node labels, use the following sample configuration:

```
config
   k8 label protocol-layer key label_key value label_value
   k8 label service-layer key label_key value label_value
   k8 label cdl-layer key label_key value label_value
   k8 label oam-layer key label_key value label_value
   end
```

Following is an example configuration of pod to node-label mapping:

```
k8 label protocol-layer key smi.cisco.com/node-type value smf-proto
exit
k8 label service-layer key vm-type value smf-svc
exit
k8 label cdl-layer key smi.cisco.com/node-type value smf-cdl
exit
k8 label oam-layer key smi.cisco.com/node-type value oam
exit
```

Deploying and Accessing cnSGW-C

This section describes how to deploy cnSGW-C and access the cnSGW-C Ops Center.

Deploying cnSGW-C

The Subscriber Microservices Infrastructure (SMI) platform is responsible for deploying and managing the cnSGW-C application and other network functions.

For information on how to deploy cnSGW-C Ops Center on bare metal servers (currently Cisco UCS-C servers) environment, see *Operating the SMI Cluster Manager on Bare Metal* section in the *Ultra Cloud Core Subscriber Microservices Infrastructure — Operations Guide*.

Accessing the cnSGW-C Ops Center

You can connect to the cnSGW-C Ops Center through SSH or the web-based CLI console.

- SSH:

  ssh admin*@ops_center_pod_ip* **-p 2024**

- Web-based console:

  1. Log in to the Kubernetes master node.
  2. Run the following command:

     kubectl get ingress <*namespace*>

     The available ingress connections get listed.
  3. Select the appropriate ingress and access the Ops Center.
  4. Access the following URL from your web browser:

     cli.<*namespace*>**-ops-center**.<*ip_address*>.**nip.io**

By default, the Day 0 configuration is loaded into the cnSGW-C.

Day 0 Configuration

To view the Day 0 configuration, run the following command.

show running-config

The following is a sample Day 0 configuration:

```
system mode shutdown
helm default-repository base-repos
helm repository base-repos
 url https://charts.209.165.201.1.nip.io/ccg.2021.01.0.i60
exit
k8s name          2nd-a18-kub-cluster
k8s namespace     cn-cn3
k8s nf-name       smf
k8s registry      docker.209.165.201.1.nip.io/ccg.2021.01.0.i60
```

```
k8s single-node   false
k8s use-volume-claims false
k8s ingress-host-name 209.165.201.2.nip.io
k8s nodes 2nd-a18-kub-cluster-master-11
 node-type   master
 worker-type master
exit
k8s nodes 2nd-a18-kub-cluster-master-22
 node-type   master
 worker-type master
exit
k8s nodes 2nd-a18-kub-cluster-master-33
 node-type   master
 worker-type master
exit
aaa authentication users user admin
 uid        1117
 gid        1117
 password   $1$XNGJOr.C$iZZvQbNfmPN15qG4GpQa8/
 ssh_keydir /tmp/admin/.ssh
 homedir    /tmp/admin
exit
aaa ios level 0
 prompt "\h> "
exit
aaa ios level 15
 prompt "\h# "
exit
aaa ios privilege exec
 level 0
  command action
  exit
  command autowizard
  exit
  command enable
  exit
  command exit
  exit
  command help
  exit
  command startup
  exit
 exit
 level 15
  command configure
  exit
 exit
exit
nacm write-default deny
nacm groups group LI
 user-name [ liadmin ]
exit
nacm groups group admin
 user-name [ admin ]
exit
nacm rule-list admin
 group [ admin ]
 rule li-deny-tap
  module-name       lawful-intercept
  path              /lawful-intercept
  access-operations *
  action            deny
 exit
 rule li-deny-clear
```

```
  module-name       tailf-mobile-smf
  path              /clear/lawful-intercept
  access-operations *
  action            deny
 exit
 rule any-access
  action permit
 exit
exit
nacm rule-list confd-api-manager
 group [ confd-api-manager ]
 rule any-access
  action permit
 exit
exit
nacm rule-list ops-center-security
 group [ * ]
 rule change-self-password
  module-name       ops-center-security
  path              /smiuser/change-self-password
  access-operations exec
  action            permit
 exit
 rule smiuser
  module-name       ops-center-security
  path              /smiuser
  access-operations exec
  action            deny
 exit
exit
nacm rule-list lawful-intercept
 group [ LI ]
 rule li-accept-tap
  module-name       lawful-intercept
  path              /lawful-intercept
  access-operations *
  action            permit
 exit
 rule li-accept-clear
  module-name       tailf-mobile-smf
  path              /clear/lawful-intercept
  access-operations *
  action            permit
 exit
exit
nacm rule-list any-group
 group [ * ]
 rule li-deny-tap
  module-name       lawful-intercept
  path              /lawful-intercept
  access-operations *
  action            deny
 exit
 rule li-deny-clear
  module-name       tailf-mobile-smf
  path              /clear/lawful-intercept
  access-operations *
  action            deny
 exit
exit
```

Loading Day 1 Configuration

The cnSGW-C configuration is provided using the Ops Center infrastructure. To load the Day 1 configuration, run the following command:

ssh admin*@ops_center_pod_ip* **-p 2024** *Day1config.cli*

Note The Day1config.cli, on page 27 file contains the necessary parameters required for the Day 1 configuration.

Alternatively, you can copy the configuration and paste it in the cnSGW-C Ops Center CLI to load the Day 1 configuration.

```
config
  <Paste the Day 1 configuration here>
  commit
  end
```

Day1config.cli

The following is a sample `Day1config.cli` file, which contains the Day 1 configuration for the cnSGW-C.

```
ipam
instance 1
source local
address-pool poolv4
vrf-name ISP
tags
dnn intershat
dnn starent.com
exit
ipv4
split-size
per-cache 1024
per-dp    256
exit
address-range 209.165.200 209.165.200.224
exit
exit
address-pool poolv4DNN2
vrf-name ISP
tags
dnn intershat1
exit
ipv4
split-size
per-cache 1024
per-dp    256
exit
address-range 209.165.100 209.165.201.0
exit
exit
address-pool poolv4DNN3
static
vrf-name ISP
tags
dnn intershat2
```

```
exit
ipv4
split-size
per-cache 512
per-dp    512
exit
address-range 209.165.202 209.165.202.128
exit
ipv6
prefix-ranges
split-size
per-cache 8192
per-dp    8192
exit
prefix-range 2002:db0:: length 48
exit
exit
exit
address-pool poolv4vDNN
vrf-name ISP
tags
dnn intershat1
exit
ipv4
split-size
per-cache 1024
per-dp    256
exit
address-range 209.165.200 209.165.202.128
exit
exit
address-pool poolv6
vrf-name ISP
tags
dnn intershat
exit
ipv6
prefix-ranges
split-size
per-cache 8192
per-dp    1024
exit
prefix-range 2001:db0:: length 48
exit
exit
exit
address-pool poolv6DNN2
vrf-name ISP
tags
dnn intershat1
exit
ipv6
prefix-ranges
split-size
per-cache 8192
per-dp    1024
exit
prefix-range 2001:ef0:: length 48
exit
exit
exit
address-pool poolv6vDNN
vrf-name ISP
tags
```

```
dnn intershat1
exit
ipv6
prefix-ranges
split-size
per-cache 8192
per-dp    1024
exit
prefix-range 2001:ab0:: length 48
exit
exit
exit
exit
cdl deployment-model small
cdl zookeeper replica 1
cdl datastore session
 slice-names 1
index map    1
index write-factor 1
slot replica 1
slot map     1
slot write-factor 1
exit
cdl kafka replica 1
etcd replicas 1
instances instance 1
 slice-name 1
 system-id  DCNAME001
 cluster-id CLUSTER0001
exit
local-instance instance 1
instance instance-id 1
endpoint sbi
replicas 1
vip-ip 209.165.201.3 vip-port 1234

  interface nrf
   loopbackPort 9001
   sla response 1000
   sla procedure 1000
   vip-ip 209.165.201.3 vip-port 9002 offline
  exit
  interface n11
   loopbackPort 9011
   sla response 1000
   sla procedure 1000
   vip-ip 209.165.201.3 vip-port 8090
  exit
  interface n7
   loopbackPort 9007
   sla response 1000
   sla procedure 1000
   vip-ip 209.165.201.3 vip-port 8090
  exit
  interface n10
   loopbackPort 9010
   sla response 1000
   sla procedure 1000
   vip-ip 209.165.201.3 vip-port 8090
  exit
  interface n40
   loopbackPort 9040
   sla response 1000
   sla procedure 1000
```

```
   vip-ip 209.165.201.3 vip-port 8090
  exit


exit
endpoint li
replicas 1
vip-ip 209.165.201.3
exit
endpoint nodemgr
replicas 1
nodes    1
exit
endpoint gtp
replicas 1
interface s5
vip-ip 209.165.200.225
exit
interface s2b
vip-ip 209.165.200.225
exit
interface s5e
vip-ip 209.165.201.3
exit
interface s11
vip-ip 209.165.200.226
exit
exit
endpoint pfcp
replicas 1
enable-cpu-optimization true
interface sxa
heartbeat
interval               5
retransmission-timeout 3
max-retransmissions    5
exit
interface n4
heartbeat
interval               0
retransmission-timeout 3
max-retransmissions    5
exit
exit
exit
#endpoint radius-dns
#replicas 1
#vip-ip 209.165.201.3
#interface radius-client
#vip-ip 209.165.201.3
#exit
#exit
endpoint service
replicas 1
nodes 1
exit
endpoint protocol
vip-ip 209.165.201.3
replicas 1
interface n4
vip-ip 209.165.200.225
exit
interface sxa
vip-ip 209.165.201.3
```

```
exit
exit
endpoint sgw-service
replicas 1
node 1
exit
exit
logging level application debug
logging level transaction debug
logging level tracing debug
logging name infra.config.core level application trace
logging name infra.config.core level transaction trace
logging name infra.config.core level tracing off
logging name infra.message_log.core level transaction trace
deployment
 model small
 app-name     SMF
 cluster-name Local
 dc-name      DC
exit
k8 label protocol-layer key disktype value ssd
#k8 label service-layer key radnaik_key value mine
#k8 label service-layer key smi.cisco.com/node-type value oam
exit
system mode running
helm default-repository cn
helm repository cn
#access-token smf-deployer.gen:Mitg_123
#access-token dev-deployer.gen:Mitg_123
#access-token
dev-deployer.gen:AKCp5ekcXA7TknM9DbLASNBw4jwVEsx9Z9WpQwEvCvCQ2mJhLymcz6BfbH38YJiWC6fn1cKmw
access-token
smf-deployer.gen:AKCp5ekcX7DcBhuAmMZYfGLaHvH3E4Syr9TQDp1gjzcSjYrqsrGbXSYs5X2XYij3d9n9VfWQe
#url
https://engci-maven-master.cisco.com/artifactory/smi-fuse-internal-snapshot/mobile-cnat-cn/cn-products/dev-cn-stage
url
https://engci-maven-master.cisco.com/artifactory/smi-fuse-internal-snapshot/mobile-cnat-cn/cn-products/dev-cn-stage
exit
profile nf-client nf-type udm
udm-profile UP1
locality LOC1
priority 30
service name type nudm-sdm
endpoint-profile EP1
capacity   30
uri-scheme http
version
uri-version v2
exit
exit
endpoint-name EP1
primary ip-address ipv4 209.165.201.3
primary ip-address port 8001
exit
exit
exit
service name type nudm-uecm
endpoint-profile EP1
capacity   30
uri-scheme http
endpoint-name EP1
primary ip-address ipv4 209.165.201.3
primary ip-address port 8001
exit
```

```
exit
exit
service name type nudm-ee
endpoint-profile EP1
capacity       30
api-uri-prefix PREFIX
api-root       ROOT
uri-scheme     http
endpoint-name EP1
priority 56
primary ip-address ipv4 209.165.201.3
primary ip-address port 8001
exit
exit
exit
exit
exit
exit
profile nf-client nf-type pcf
pcf-profile PP1
locality LOC1
priority 30
service name type npcf-am-policy-control
endpoint-profile EP1
capacity   30
uri-scheme http
endpoint-name EP1
priority 56
primary ip-address ipv4 209.165.201.3
primary ip-address port 8003
exit
exit
exit
service name type npcf-smpolicycontrol
endpoint-profile EP1
capacity   30
uri-scheme http
endpoint-name EP1
priority 56
primary ip-address ipv4 209.165.201.3
primary ip-address port 8003
exit
exit
exit
exit
exit
exit
profile nf-client nf-type amf
amf-profile AP1
locality LOC1
priority 30
service name type namf-comm
endpoint-profile EP2
capacity   30
uri-scheme http
endpoint-name EP1
priority 56
primary ip-address ipv4 209.165.201.3
primary ip-address port 8002
exit
exit
exit
exit
exit
```

```
exit
profile nf-client nf-type chf
chf-profile CP1
locality LOC1
priority 30
service name type nchf-convergedcharging
endpoint-profile EP1
capacity   30
uri-scheme http
version
uri-version v2
exit
exit
endpoint-name EP1
priority 56
primary ip-address ipv4 209.165.201.3
primary ip-address port 8004
exit
exit
exit
exit
exit
chf-profile CP2
locality LOC1
priority 31
service name type nchf-convergedcharging
endpoint-profile EP1
capacity   30
uri-scheme http
version
uri-version v2
exit
exit
endpoint-name EP1
priority 56
primary ip-address ipv4 209.165.201.3
primary ip-address port 9040
exit
exit
exit
exit
exit
exit
profile nf-pair nf-type UDM
nrf-discovery-group udmdiscovery
locality client  LOC1
locality preferred-server LOC1
locality geo-server GEO
exit
profile nf-pair nf-type AMF
nrf-discovery-group udmdiscovery
locality client  LOC1
locality preferred-server LOC1
locality geo-server GEO
exit
profile nf-pair nf-type PCF
nrf-discovery-group udmdiscovery
locality client  LOC1
locality preferred-server LOC1
locality geo-server GEO
exit
profile nf-pair nf-type UPF
nrf-discovery-group udmdiscovery
locality client  LOC1
```

```
locality preferred-server LOC1
locality geo-server GEO
exit
profile nf-pair nf-type CHF
nrf-discovery-group udmdiscovery
locality client  LOC1
locality preferred-server LOC1
locality geo-server GEO
exit
profile nf-client-failure nf-type udm
profile failure-handling FH4
service name type nudm-sdm
message type UdmSdmGetUESMSubscriptionData
status-code httpv2 403
retry  3
action retry-and-ignore
exit
status-code httpv2 404
action continue
exit
status-code httpv2 413
retry  3
action retry-and-continue
exit
status-code httpv2 501
retry  3
action retry-and-terminate
exit
status-code httpv2 503
action terminate
exit
status-code httpv2 504
retry  3
action retry-and-terminate
exit
exit
message type UdmSdmSubscribeToNotification
status-code httpv2 403
retry  3
action retry-and-ignore
exit
status-code httpv2 404
action continue
exit
status-code httpv2 413
retry  3
action retry-and-continue
exit
status-code httpv2 501
retry  3
action retry-and-terminate
exit
status-code httpv2 503
action terminate
exit
status-code httpv2 504
retry  3
action retry-and-terminate
exit
exit
exit
service name type nudm-uecm
message type UdmUecmRegisterSMF
status-code httpv2 403
```

```
retry  3
action retry-and-ignore
exit
status-code httpv2 404
action continue
exit
status-code httpv2 413
retry  3
action retry-and-continue
exit
status-code httpv2 501
retry  3
action retry-and-terminate
exit
status-code httpv2 503
action terminate
exit
status-code httpv2 504
retry  3
action retry-and-terminate
exit
exit
exit
exit
exit
profile nf-client-failure nf-type pcf
profile failure-handling FH1
service name type npcf-smpolicycontrol
message type PcfSmpolicycontrolCreate
status-code httpv2 0
action retry-and-ignore
exit
status-code httpv2 400
action continue
exit
status-code httpv2 403
action retry-and-ignore
exit
status-code httpv2 404
action terminate
exit
status-code httpv2 500
retry  2
action retry-and-ignore
exit
status-code httpv2 503
retry  2
action retry-and-continue
exit
exit
message type PcfSmpolicycontrolUpdate
status-code httpv2 0
action retry-and-ignore
exit
status-code httpv2 400
action continue
exit
status-code httpv2 403
action retry-and-ignore
exit
status-code httpv2 404
action terminate
exit
status-code httpv2 500
```

```
retry  2
action retry-and-ignore
exit
status-code httpv2 503
retry  2
action retry-and-continue
exit
exit
message type PcfSmpolicycontrolDelete
status-code httpv2 0
action retry-and-ignore
exit
status-code httpv2 400
action continue
exit
status-code httpv2 403
action retry-and-ignore
exit
status-code httpv2 404
action terminate
exit
status-code httpv2 500
retry  2
action retry-and-ignore
exit
status-code httpv2 503
retry  2
action retry-and-continue
exit
exit
exit
exit
exit
profile nf-client-failure nf-type chf
profile failure-handling FH2
service name type nchf-convergedcharging
message type ChfConvergedchargingCreate
status-code httpv2 0
action continue
exit
status-code httpv2 400
retry  3
action retry-and-terminate
exit
status-code httpv2 403
retry  3
action retry-and-ignore
exit
status-code httpv2 404
retry  3
action retry-and-terminate
exit
status-code httpv2 500
action continue
exit
status-code httpv2 503
action terminate
exit
status-code httpv2 504
action continue
exit
exit
message type ChfConvergedchargingUpdate
status-code httpv2 0
```

```
action continue
exit
status-code httpv2 400
retry  3
action retry-and-terminate
exit
status-code httpv2 403
retry  3
action retry-and-ignore
exit
status-code httpv2 404
retry  3
action retry-and-terminate
exit
status-code httpv2 500
action continue
exit
status-code httpv2 503
action terminate
exit
status-code httpv2 504
action continue
exit
exit
message type ChfConvergedchargingDelete
status-code httpv2 0
action continue
exit
status-code httpv2 400
retry  3
action retry-and-terminate
exit
status-code httpv2 403
retry  3
action retry-and-ignore
exit
status-code httpv2 404
retry  3
action retry-and-terminate
exit
status-code httpv2 500
action continue
exit
status-code httpv2 503
action terminate
exit
status-code httpv2 504
action continue
exit
exit
exit
exit
exit
profile sgw sgw1
 locality          LOC2
 fqdn              cisco.com.apn.epc.mnc456.mcc123
 #subscriber-policy polSub
exit
profile smf smf1
node-id       abcdef
locality      LOC1
fqdn          cisco.com.apn.epc.mnc456.mcc123
allowed-nssai [ slice1 ]
plmn-id mcc 123
```

```
plmn-id mnc 456
service name nsmf-pdu
type               pdu-session
schema             http
service-id         1
version            1.Rn.0.0
http-endpoint base-url http://smf-service
icmpv6-profile     icmpprf1
compliance-profile comp1
access-profile     access1
subscriber-policy  polSub
exit
exit
profile sgw sgw1
locality      LOC2
fqdn          cisco.com.apn.epc.mnc456.mcc123
plmn-id mcc 123
plmn-id mnc 456
#subscriber-policy  polSub
exit
profile dnn starent.com
 network-element-profiles chf chf1
 network-element-profiles amf amf1
 network-element-profiles pcf pcf1
 network-element-profiles udm udm1
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:47
 ssc-mode 2 allowed [ 3 ]
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn starent.com
 #dcnr true
exit

profile dnn default-profile
 network-element-profiles chf chf1
 network-element-profiles amf amf1
 network-element-profiles pcf pcf1
 network-element-profiles udm udm1
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:47
 ssc-mode 2 allowed [ 3 ]
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn starent.com
 #dcnr true
exit

profile dnn intershat
network-element-profiles chf chf1
network-element-profiles amf amf1
network-element-profiles pcf pcf1
network-element-profiles udm udm1
charging-profile chgprf1
virtual-mac      b6:6d:47:47:47:47
ssc-mode 2 allowed [ 3 ]
session type IPV4 allowed [ IPV6 IPV4V6 ]
upf apn intershat
dcnr             true
exit
profile dnn intershat1
network-element-profiles chf chf1
network-element-profiles amf amf1
network-element-profiles pcf pcf1
network-element-profiles udm udm1
charging-profile chgprf1
```

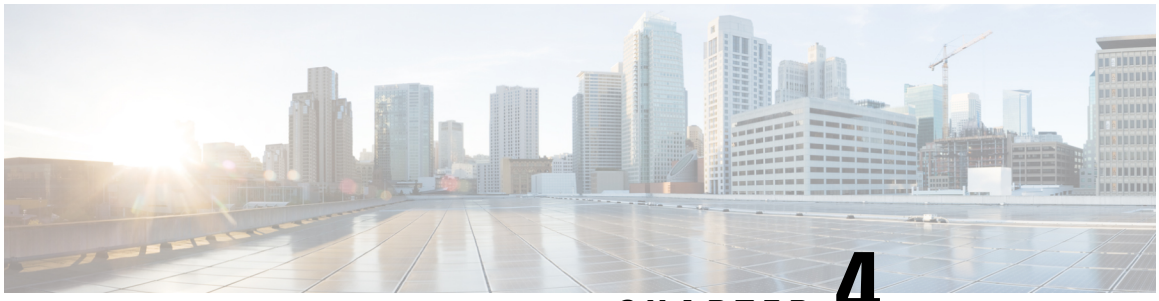
```
virtual-mac      b6:6d:47:47:47:48
pcscf-profile    PCSCF_Prof_2
ssc-mode 1
session type IPV4
exit
profile dnn intershat2
network-element-profiles chf chf
network-element-profiles amf amf
network-element-profiles pcf pcf
network-element-profiles udm udm
charging-profile chgprf1
virtual-mac      b6:6d:47:47:47:49
ssc-mode 2 allowed [ 3 ]
session type IPV4 allowed [ IPV6 IPV4V6 ]
upf apn intershat2
exit
profile qos abc
ambr ul "250 Kbps"
ambr dl "500 Kbps"
qi5      7
arp priority-level 14
arp preempt-cap NOT_PREEMPT
arp preempt-vuln PREEMPTABLE
priority 120
max data-burst 2000
exit
profile failure-handling FH1
interface pfcp message N4SessionEstablishmentReq
cause-code pfcp-entity-in-congestion action retry-terminate max-retry 2
cause-code system-failure action terminate
cause-code service-not-supported action terminate
cause-code no-resource-available action retry-terminate max-retry 3
cause-code no-response-received action retry-terminate max-retry 1
cause-code reject action terminate
exit
interface pfcp message N4SessionModificationReq
cause-code mandatory-ie-incorrect action terminate
cause-code session-ctx-not-found action terminate
cause-code reject action terminate
exit
exit
profile failure-handling gtp1
interface gtpc message S5S8CreateBearerReq
cause-code temp-fail
action retry timeout 1000 max-retry 2
exit
exit
interface gtpc message S5S8UpdateBearerReq
cause-code temp-fail
action retry timeout 1000 max-retry 2
exit
exit
interface gtpc message S5S8DeleteBearerReq
cause-code temp-fail
action retry timeout 1000 max-retry 2
exit
exit
exit
profile network-element amf amf1
nf-client-profile        AP1
failure-handling-profile FH3
query-params [ dnn ]
exit
profile network-element pcf pcf1
```

```
nf-client-profile        PP1
failure-handling-profile FH1
query-params [ dnn ]
rulebase-prefix          cbn#
predefined-rule-prefix   crn#
exit
profile network-element udm udm1
nf-client-profile        UP1
failure-handling-profile FH4
query-params [ dnn ]
exit
profile network-element upf upf226
node-id upf226@sgw.com
n4-peer-address ipv4 209.165.201.4
n4-peer-port 8805
dnn-list     [ intershat intershat1 intershat2 cisco.com starent.com ]
capacity     2000
priority     10
exit
profile network-element upf upf1
node-id upf1@sgw.com
n4-peer-address ipv4 209.165.201.5
n4-peer-port 8805
dnn-list     [ intershat intershat1 intershat2 cisco.com starent.com ]
capacity     2000
priority     10
exit
profile network-element upf upf2
node-id upf2@sgw.com
n4-peer-address ipv4 209.165.201.6
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 cisco.com starent.com ]
capacity     2000
priority     1
exit
profile network-element upf upf76
node-id upf3@sgw.com
n4-peer-address ipv4 209.165.201.7
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     1000
priority     10
exit
profile network-element upf upf70
node-id upf4@sgw.com
n4-peer-address ipv4 209.165.201.8
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     1000
priority     10
exit
profile network-element upf upf71
node-id upf5@sgw.com
n4-peer-address ipv4 209.165.201.9
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     1000
priority     10
exit
profile network-element upf upf72
n4-peer-address ipv4 209.165.201.10
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
```

```
priority     10
exit
profile network-element upf upf79
n4-peer-address ipv4 209.165.201.11
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element upf upf131
n4-peer-address ipv4 209.165.201.12
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element upf upf132
n4-peer-address ipv4 209.165.201.13
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element upf upf133
n4-peer-address ipv4 209.165.201.14
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element upf upf134
n4-peer-address ipv4 209.165.201.15
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element upf upf135
n4-peer-address ipv4 209.165.201.16
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element upf upf136
n4-peer-address ipv4 209.165.201.17
n4-peer-port 8805
dnn-list     [ intershat1 intershat2 starent.com cisco.com ]
capacity     2000
priority     10
exit
profile network-element chf chf1
nf-client-profile         CP1
failure-handling-profile  FH2
query-params [ dnn ]
nf-client-profile-offline CP2
exit
profile network-element chf chgser1
exit
profile compliance comp1
service nsmf-pdusession
version uri v1
version full 1.0.0
version spec 15.4.0
```

```
exit
service namf-comm
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service n1
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service n2
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service nudm-sdm
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service nudm-uecm
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service nnrf-disc
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service nnrf-nfm
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service npcf-smpolicycontrol
version uri v1
version full 1.0.0
version spec 15.4.0
exit
service nchf-convergedcharging
version uri v1
version full 1.0.0
version spec 15.3.0
exit
exit
profile upf-group group1
failure-profile FH1
exit
profile access access1
n26 idft enable timeout 15
n2 idft enable timeout 15
gtpc gtpc-failure-profile gtp1
exit
profile icmpv6 icmpprf1
options virtual-mac b6:6d:57:45:45:45
exit
profile charging chgprf1
method [ offline ]
exit
profile charging-characteristics 1
charging-profile chgprf1
exit
```

```
nssai name slice1
sst 2
sdt Abf123
dnn [ dnn1 intershat intershat1 intershat2 ]
exit
policy subscriber polSub
precedence 1
sst              02
sdt              Abf123
serving-plmn mcc 123
serving-plmn mnc 456
supi-start-range 100000000000001
supi-stop-range  999999999999999
gpsi-start-range 1000000000
gpsi-stop-range  9999999999
operator-policy  opPol1
exit
precedence 511
operator-policy defOprPol1
exit
exit
policy operator defOprPol1
policy dnn         defPolDnn
policy network-capability nc1
exit
policy operator opPol1
policy dnn         polDnn
policy network-capability nc1
exit
policy dnn defPolDnn
profile default-profile
dnn dnn2 profile profile2
dnn intershat profile intershat
dnn intershat1 profile intershat1
dnn starent.com profile starent.com
exit
policy dnn polDnn
profile default-profile
dnn dnn2 profile profile2
dnn intershat profile intershat
dnn intershat1 profile intershat1
dnn intershat2 profile intershat2
dnn starent.com profile starent.com
exit
policy network-capability nc1
nw-support-local-address-tft true
exit
nacm groups group LI2
user-name [ liadmin2 ]
exit
nacm groups group LI3
user-name [ liadmin3 ]
exit
nacm groups group admin
user-name [ admin ]
exit
commit
end
```



CHAPTER 4

cnSGW-C Rolling Software Update

- Feature Summary and Revision History, on page 45
- Introduction, on page 45
- Updating cnSGW-C, on page 46
- Rolling Upgrade Optimization, on page 57

Feature Summary and Revision History

Summary Data

Table 5: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Not Applicable
Related Documentation	Not Applicable

Revision History

Table 6: Revision History

Revision Details	Release
First introduced.	2021.02.0

Introduction

The cnSGW-C has a three-tier architecture consisting of Protocol, Service, and Session. Each tier includes a set of microservices (pods) for a specific functionality. Within these tiers, there exists a Kubernetes Cluster

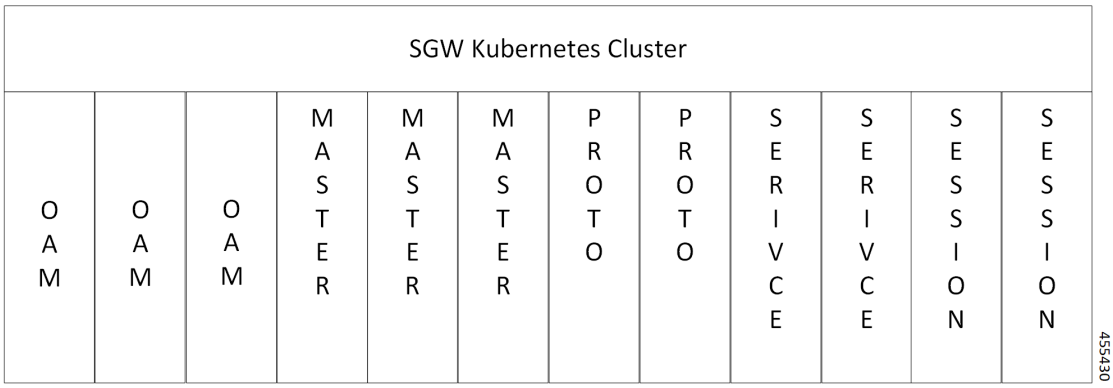
comprising of Kubernetes (K8s) master, and worker nodes (including Operation and Management (OAM) nodes).

For high availability and fault tolerance, a minimum of two K8s worker nodes are configured for each tier. You can have multiple replicas for each worker node. Kubernetes orchestrates the pods using the StatefulSets controller. The pods require a minimum of two replicas for fault tolerance.

The following figure depicts cnSGW-C K8s cluster with 12 nodes.

- Three master nodes
- Three OAM worker nodes
- Two Protocol worker nodes
- Two Service worker nodes
- Two Session (data store) worker nodes

Figure 12: cnSGW-C Kubernetes Cluster



The cnSGW-C Kubernetes cluster comprises of the following nodes:

- The OAM worker nodes host the Ops Center pods for configuration management and metrics pods for statistics and Key Performance Indicators (KPIs).
- The Protocol worker nodes host the cnSGW-C protocol-related pods for service-based interfaces (N11, N7, N10, N40) and UDP-based protocol interfaces (N4, S5/S8).
- The Service worker nodes host the cnSGW-C application-related pods that perform session management processing.
- The Session worker nodes host the database-related pods that store subscriber session data.

Updating cnSGW-C

The rolling software update is a process of updating or migrating the build from an older to a newer version or updating the patch for the prescribed deployment set of application pods.

Rolling update takes place with zero downtime by incrementally updating the pod instances with the new ones.

Note The applications must be available when new versions are expected to be deployed with the new build versions or patches.

Update Scope

The rolling update feature is supported from an older to the newer versions within the same major release.

- **Assumptions:** When updating, it is assumed that the following has not been changed between the versions:
  - Features supported in the old and the new versions.
  - Configuration addition, deletion, or modification of the existing CLI behavior.
  - Interface change within the peer or across the pods.
- **Recommendations:**
  - Configuration changes are not recommended during the update process.
  - All configuration changes should be done after the update process is complete.
- **Failure Handling:** The system should be downgraded manually to an older healthy build following the downgrade process for:
  - Failure during the process such as crash, and pods deployment failures.
  - Failure after the successful update such as new events or procedures.

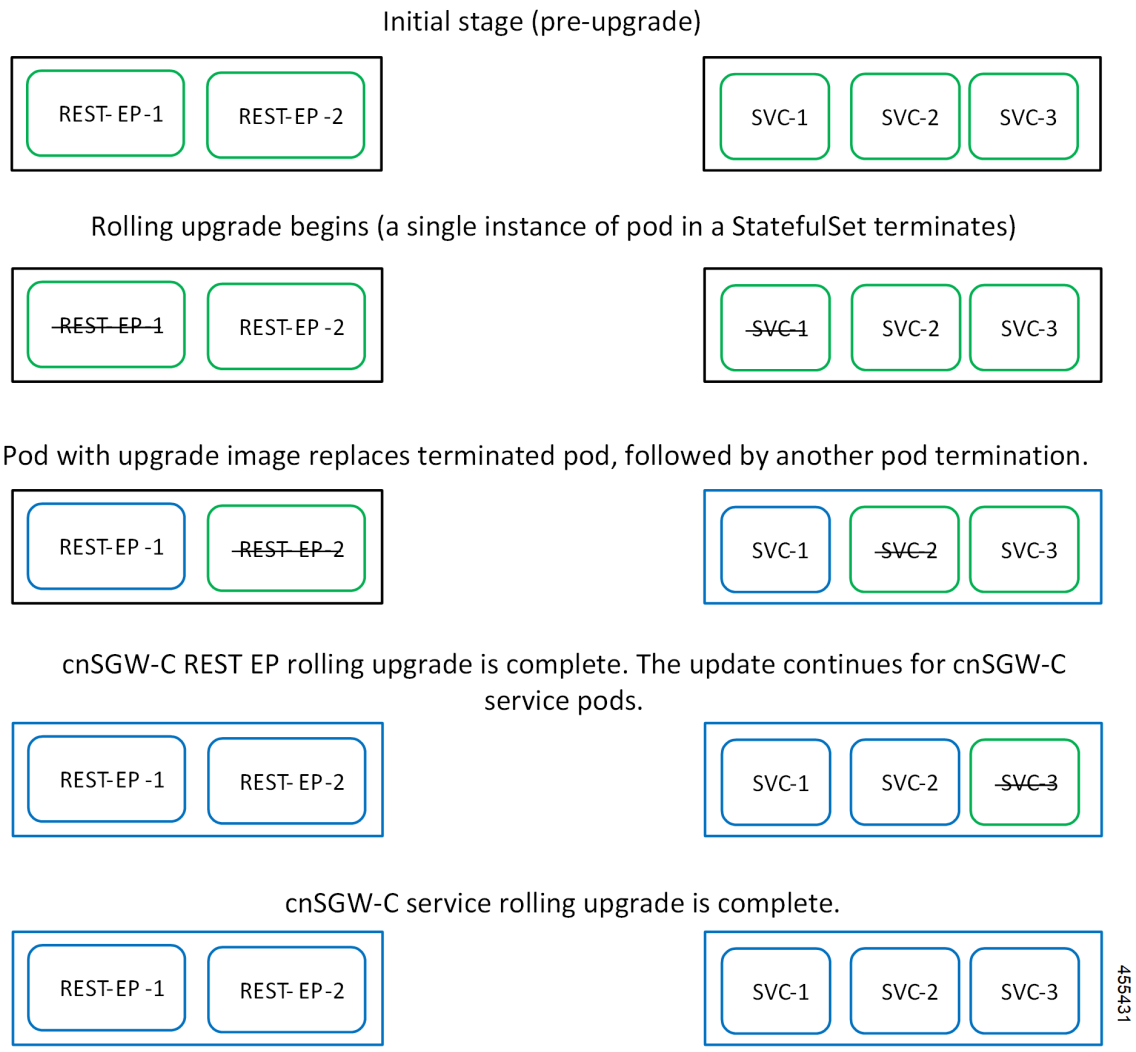
Rolling Software Update Using the SMI Cluster Manager

The cnSGW-C software update or in-service update procedure utilizes the K8s rolling strategy to update the pod images. In this strategy, the pods of a StatefulSet are updated sequentially to ensure that the ongoing process remains unaffected. Initially, a rolling update on a StatefulSet causes a single pod instance to terminate. A pod with an updated image replaces the terminated pod. This process continues until all the replicas of the StatefulSet are updated. The terminating pods exit gracefully after competing all the ongoing processes. Other in-service pods continue to receive and process the traffic to provide a seamless software update. You can control the software update process through the Ops Center CLI.

Note Each pod needs a minimum of two pods for high availability. In a worst-case scenario, the processing capacity of the pod may briefly reduce to 50% while the software update is in-progress.

The following figure illustrates a cnSGW-C rolling update for cnSGW-C REST endpoint pods (two replicas) on Protocol worker nodes along with cnSGW-C Service pods (three replicas) on Service worker nodes.

Figure 13: cnSGW-C Rolling Update



Prerequisites

The prerequisites for upgrading cnSGW-C are:

- All the nodes that include all the pods in the node that are up and running.
- A patch version of the cnSGW-C software.

Note Major versions do not support rolling update.

Important Trigger rolling update only when the CPU usage of the nodes is less than 50%.

- Intra-site HA support.

cnSGW-C Health Check

Perform a health check to ensure that all the services are running and the nodes are in the ready state.

To perform health check, use the following configuration:

- Log in to the master node and use the following configuration:

```
kubectl get pods -n smi
kubectl get nodes
kubectl get pod --all-namespaces -o wide
kubectl get pods -n cnsgw-wsp -o wide
kubectl get pods -n cee-wsp -o wide
kubectl get pods -n smi-vips -o wide
helm list
kubectl get pods -A | wc -l
```



Important Make sure that all the services are running and nodes are in the ready state before you proceed.

Backing Up the Deployment File

To create a backup configuration, logs, and deployment files, use the following configuration:

1. Log in to the SMI Cluster Manager Node as an Ubuntu user.
2. Create a new directory for deployment.

   Example:

   ```
   test@smicnsgw-cm01:~$ mkdir -p "temp_$(date +'%m%d%Y_T%H%M')" && cd "$_"
   ```

3. Back up the working files into the newly created deployment directory.
4. Untar the `cnsgw` deployment file.

   Example:

   ```
   test@smi1cnsgw01-cm01:~/temp_08072019_T1651$ tar -xzvf cnsgw.2020.01.0-1.SPA.tgz
   ./
   ./cnsgw_REL_KEY-CCO_RELEASE.cer
   ./cisco_x509_verify_release.py
   ./cnsgw.2020.01.0-1.tar
   ./cnsgw.2020.01.0-1.tar.signature.SPA
   ./cnsgw.2020.01.0-1.tar.SPA.README
   ```

5. Verify the downloaded image.

   Example:

   ```
   test@smi1cnsgw01-cm01:~/temp_08072019_T1651$ cat cnsgw.2020.01.0-1.tar.SPA.README
   ```



Important Follow the procedure mentioned in the *SPA.README* file to verify the build before proceeding to the next step.

Backing Up the Ops Center Configuration

To back up the Ops Center configurations, use the following configuration:

1. Log in to the SMI Cluster Manager node as an Ubuntu user.
2. Back up the SMI Ops Center configuration to the **/home/ubuntu/smiops.backup** file, using the following configuration:

   ```
   ssh -p <port_number> admin@$(kubectl get svc -n smi | grep
   '.*netconf.*<port_number>' | awk '{ print $4 }') "show run | nomore"
   > smiops.backup_$(date +'%m%d%Y_T%H%M')
   ```

3. Back up the CEE Ops Center configuration to the **/home/ubuntu/ceeops.backup** file, using the following configuration:

   ```
   ssh admin@<cee-vip> "show run | nomore" > ceeops.backup_$(date
   +'%m%d%Y_T%H%M')
   ```

4. Back up the cnSGW-C Ops Center configuration to the **/home/ubuntu/cnSGWops.backup** file, using the following configuration:

   ```
   ssh admin@<cnSGW-vip> "show run | nomore" > cnSGWops.backup_$(date
   +'%m%d%Y_T%H%M')
   ```

Back Up CEE and cnSGW-C Ops Center Configuration

To back up the CEE and Ops Center configuration from the master node, use the following configuration:

1. Log in to the master node as an Ubuntu user.
2. Create a directory to backup the configuration files, using the following configuration:

   ```
   mkdir backups_$(date +'%m%d%Y_T%H%M') && cd "$_"
   ```

3. Back up the cnSGW-C Ops Center configuration and verify the line count of the backup files, using the following configuration:

   ```
   ssh -p <port_number> admin@$(kubectl get svc -n $(kubectl get namespaces
    | grep -oP 'cnSGW-(\d+|\w+)') | grep <port_number> | awk '{ print $3
    }') "show run | nomore" > cnSGWops.backup_$(date +'%m%d%Y_T%H%M') &&
    wc -l cnSGWops.backup_$(date +'%m%d%Y_T%H%M')
   ```

   Example:

   ```
   ubuntu@pocnSGW-mas01:~/backups_09182019_T2141$ ssh -p 2024 admin@$(kubectl get svc -n
   $(kubectl get namespaces | grep -oP 'cnSGW-(\d+|\w+)') | grep <port_number> | awk '{
   print $3 }') "show run | nomore" > cnSGWops.backup_$(date +'%m%d%Y_T%H%M') && wc -l
   cnSGWops.backup_$(date +'%m%d%Y_T%H%M')
   admin@<ipv4address>'s password: cnSGW-OPS-PASSWORD
   334 cnSGWops.backup
   ```

4. Back up the CEE Ops Center configuration and verify the line count of the backup files, using the following configuration:

   ```
   ssh -p <port_number> admin@$(kubectl get svc -n $(kubectl get namespaces
    | grep -oP 'cee-(\d+|\w+)') | grep <port_number> | awk '{ print $3
   }') "show run | nomore" > ceeops.backup_$(date +'%m%d%Y_T%H%M') && wc
    -l ceeops.backup_$(date +'%m%d%Y_T%H%M')
   ```

   Example:

```
ubuntu@pocnSGW-mas01:~/backups_09182019_T2141$ ssh -p <port_number> admin@$(kubectl get
 svc -n $(kubectl get namespaces | grep -oP 'cee-(\d+|\w+)') | grep <port_number> | awk
 '{ print $3 }') "show run | nomore" > ceeops.backup_$(date +'%m%d%Y_T%H%M') && wc -l
ceeops.backup_$(date +'%m%d%Y_T%H%M')
admin@<ipv4address>'s password: CEE-OPS-PASSWORD
233 ceeops.backup
```

5. Move the SMI Ops Center backup file (from the SMI Cluster Manager) to the backup directory, using the following configuration:

   scp $(grep cm01 /etc/hosts | awk '{ print $1 }'):/home/ubuntu/smiops.backup_$(date +'%m%d%Y_T%H%M') .

   Example:

   ```
   ubuntu@pocnSGW-mas01:~/backups_09182019_T2141$ scp $(grep cm01 /etc/hosts | awk '{ print
    $1 }'):/home/ubuntu/smiops.backup_$(date +'%m%d%Y_T%H%M') .
   ubuntu@<ipv4address>'s password: SMI-CM-PASSWORD
   smiops.backup                                                 100% 9346    22.3MB/s
      00:00
   ```

6. Verify the line count of the backup files.

   Example:

   ```
   ubuntu@pocnSGW-mas01:~/backups_09182019_T2141$ wc -l *
     233 ceeops.backup
     334 cnSGWops.backup
     361 smiops.backup
     928 total
   ```

Staging a New cnSGW-C Image

To stage a new cnSGW-C image before initiating the update, use the following configuration:

1. Download and verify the new cnSGW-C image.
2. Log in to the SMI Cluster Manager node as an Ubuntu user.
3. Copy the image to the **uploads** directory.

   sudo mv <cnSGW_new_image.tar> /data/software/uploads

Note The SMI uses the new image present in the **uploads** directory to update.

4. Verify whether the image is picked up by the SMI for processing from the **uploads** directory.

   sleep 30; ls /data/software/uploads

   Example:

   ```
   ubuntu@pocnSGW-cm01:~/temp_08072019_T1651$ sleep 30; ls /data/software/uploads
   ubuntu@pocnSGW-cm01:~/temp_08072019_T1651$
   ```

5. Verify whether the images were successfully picked up and processed.

   Example:

   ```
   auser@unknown:$ sudo du -sh /data/software/packages/*
   1.6G /data/software/packages/cee.2019.07
   ```

```
5.3G /data/software/packages/cnSGW.2019.08-04
16K /data/software/packages/sample
```

Note The SMI must unpack the images into the **packages** directory successfully to complete the staging.

Triggering the Rolling Software Upgrade

cnSGW-C utilizes the SMI Cluster Manager to perform a rolling software update.

To update cnSGW-C using SMI Cluster Manager, use the following configurations:

Important Before you begin, ensure that cnSGW-C is up and running with the current version of the software.

1. Log in to the SMI Cluster Manager Ops Center.
2. Download the latest tarall from the URL.

   software-packages download *url*

   NOTES:

   - **software-packages download** *url*—Specifies the software packages to be downloaded through HTTP/HTTPS.

   Example:

   ```
   SMI Cluster Manager# software-packages download <url>
   ```

3. Verify whether the tarall is loaded.

   software-packages list

   NOTES:

   - **software-packages list** —Specifies the list of available software packages.

   Example:

   ```
   SMI Cluster Manager# software-packages list
   [ cnSGW-2019-08-21 ]
   [ sample ]
   ```

4. Update the product repository URL with the latest version of the product chart.

Note If the repository URL contains multiple versions, the Ops Center automatically selects the latest version.

```
configure
  cluster cluster_name
   ops-centers app_name cnSGW_instance_name
       repository url
```

```
  exit
exit
```

Example:

```
SMI Cluster Manager# config
SMI Cluster Manager(config)# clusters test2
SMI Cluster Manager(config-clusters-test2)# ops-centers cnSGW data
SMI Cluster Manager(config-ops-centers-cnSGW/data)# repository <url>
SMI Cluster Manager(config-ops-centers-cnSGW/data)# exit
SMI Cluster Manager(config-clusters-test2)# exit
```

5. To update to the latest version of the product chart, run the **cluster sync** command using the following command:

   clusters *cluster_name* **actions sync run**

   Example:

   ```
   SMI Cluster Manager# clusters test2 actions sync run
   ```

NOTES:

- **cluster** —Specifies the K8s cluster.
- *cluster_name* —Specifies the name of the cluster.
- **ops-centers** *app_name instance_name* —Specifies the product Ops Center and instance. *app_name* is the application name. *instance_name* is the name of the instance.
- **repository** *url*—Specifies the local registry URL for downloading the charts.
- **actions** —Specifies the actions performed on the cluster.
- **sync run** —Triggers the cluster synchronization.



Important

- The cluster synchronization updates the cnSGW-C Ops Center, which in turn updates the application pods (through **helm sync** command) one at a time automatically.
- When you trigger rolling upgrade on a specific pod, the cnSGW-C avoids routing new calls to that pod.
- The cnSGW-C honors in-progress call by waiting for 30 seconds before restarting the pod where rolling upgrade is initiated. Also, the cnSGW-C establishes all the in-progress calls completely within 30 seconds during the upgrade period (maximum call-setup time is 10 seconds).

Monitoring the Update Procedure

To monitor the status update through SMI Cluster Manager Ops Center, use the following configurations:

```
config
  clusters cluster_name actions sync run debug true
  clusters cluster_name actions sync logs
  monitor sync-logs cluster_name
  clusters cluster_name actions sync status
  exit
```

Example:

```
SMI Cluster Manager# clusters test1 actions sync run
SMI Cluster Manager# clusters test1 actions sync run debug true
SMI Cluster Manager# clusters test1 actions sync logs
SMI Cluster Manager# monitor sync-logs test1
SMI Cluster Manager# clusters test1 actions sync status
```

NOTES:

- **clusters** *cluster_name*—Specifies the information about the nodes to be deployed. *cluster_name* is the name of the cluster.
- **actions**—Specifies the actions performed on the cluster.
- **sync run**—Triggers the cluster synchronization.
- **sync logs**—Shows the current cluster synchronization logs.
- **sync status** —Shows the current status of the cluster synchronization.
- **debug true**—Enters the debug mode.
- **monitor sync logs**—Monitors the cluster synchronization process.



Important You can view the pod details after the upgrade through the CEE Ops Center. For more information on pod details, see Viewing the Pod Details, on page 54 section.

Viewing the Pod Details

To view the details of the current pods through CEE Ops Center, use the following command in the CEE Ops Center CLI:

`cluster pods` *`instance_name pod_name`* **`detail`**

NOTES:

- **cluster pods**—Specifies the current pods in the cluster.
- *instance_name*—Specifies the name of the instance.
- *pod_name*—Specifies the name of the pod.
- **detail**—Displays the details of the specified pod.

The following example displays the details of the pod named *alertmanager-0* in the *cnSGW-data* instance.

Example:

```
cee# cluster pods cnSGW-data alertmanager-0 detail
details apiVersion: "v1"
kind: "Pod"
metadata:
  annotations:
    alermanager.io/scrape: "true"
    cni.projectcalico.org/podIP: "<ipv4address/subnet>"
    config-hash: "5532425ef5fd02add051cb759730047390b1bce51da862d13597dbb38dfbde86"
  creationTimestamp: "2020-02-26T06:09:13Z"
  generateName: "alertmanager-"
  labels:
    component: "alertmanager"
```

```
    controller-revision-hash: "alertmanager-67cdb95f8b"
    statefulset.kubernetes.io/pod-name: "alertmanager-0"
  name: "alertmanager-0"
  namespace: "cnSGW"
  ownerReferences:
  - apiVersion: "apps/v1"
    kind: "StatefulSet"
    blockOwnerDeletion: true
    controller: true
    name: "alertmanager"
    uid: "82a11da4-585e-11ea-bc06-0050569ca70e"
  resourceVersion: "1654031"
  selfLink: "/api/v1/namespaces/cnSGW/pods/alertmanager-0"
  uid: "82aee5d0-585e-11ea-bc06-0050569ca70e"
spec:
  containers:
  - args:
    - "/alertmanager/alertmanager"
    - "--config.file=/etc/alertmanager/alertmanager.yml"
    - "--storage.path=/alertmanager/data"
    - "--cluster.advertise-address=$(POD_IP):6783"
    env:
    - name: "POD_IP"
      valueFrom:
        fieldRef:
          apiVersion: "v1"
          fieldPath: "status.podIP"
    image: "<path_to_docker_image>"
    imagePullPolicy: "IfNotPresent"
    name: "alertmanager"
    ports:
    - containerPort: 9093
      name: "web"
      protocol: "TCP"
    resources: {}
    terminationMessagePath: "/dev/termination-log"
    terminationMessagePolicy: "File"
    volumeMounts:
    - mountPath: "/etc/alertmanager/"
      name: "alertmanager-config"
    - mountPath: "/alertmanager/data/"
      name: "alertmanager-store"
    - mountPath: "/var/run/secrets/kubernetes.io/serviceaccount"
      name: "default-token-kbjnx"
      readOnly: true
  dnsPolicy: "ClusterFirst"
  enableServiceLinks: true
  hostname: "alertmanager-0"
  nodeName: "for-smi-cdl-1b-worker94d84de255"
  priority: 0
  restartPolicy: "Always"
  schedulerName: "default-scheduler"
  securityContext:
    fsGroup: 0
    runAsUser: 0
  serviceAccount: "default"
  serviceAccountName: "default"
  subdomain: "alertmanager-service"
  terminationGracePeriodSeconds: 30
  tolerations:
  - effect: "NoExecute"
    key: "node-role.kubernetes.io/oam"
    operator: "Equal"
    value: "true"
```

```
  - effect: "NoExecute"
    key: "node.kubernetes.io/not-ready"
    operator: "Exists"
    tolerationSeconds: 300
  - effect: "NoExecute"
    key: "node.kubernetes.io/unreachable"
    operator: "Exists"
    tolerationSeconds: 300
  volumes:
  - configMap:
      defaultMode: 420
      name: "alertmanager"
    name: "alertmanager-config"
  - emptyDir: {}
    name: "alertmanager-store"
  - name: "default-token-kbjnx"
    secret:
      defaultMode: 420
      secretName: "default-token-kbjnx"
status:
  conditions:
  - lastTransitionTime: "2020-02-26T06:09:02Z"
    status: "True"
    type: "Initialized"
  - lastTransitionTime: "2020-02-26T06:09:06Z"
    status: "True"
    type: "Ready"
  - lastTransitionTime: "2020-02-26T06:09:06Z"
    status: "True"
    type: "ContainersReady"
  - lastTransitionTime: "2020-02-26T06:09:13Z"
    status: "True"
    type: "PodScheduled"
  containerStatuses:
  - containerID: "docker://821ed1a272d37e3b4c4c9c1ec69b671a3c3fe6eb4b42108edf44709b9c698ccd"

    image: "<path_to_docker_image>"
    imageID: "docker-pullable://<path_to_docker_image>"
    lastState: {}
    name: "alertmanager"
    ready: true
    restartCount: 0
    state:
      running:
        startedAt: "2020-02-26T06:09:05Z"
  hostIP: "<host_ipv4address>"
  phase: "Running"
  podIP: "<pod_ipv4address>"
  qosClass: "BestEffort"
  startTime: "2020-02-26T06:09:02Z"
cee#
```

Rolling Software Update on Non-SMI Cluster

To configure the helm repository, use the following configuration:

- Log in to cnSGW-C Ops Center and use the following configuration:

```
config
 helm default-repository cn
 helm repository cn
 access-token
smf-deployer.gen:AKCp5ekcX7DcBhuAmMZYfGLaHvH3E4Syr9TQDp1gjzcSjYrqsrGbXSYs5X2XYij3d9n9VfWQe
```

```
 url <old-build/new-build>
exit
```

Validating the Update

The health check, current helm charts, and subscriber/peer/session information help in understanding whether the rolling update process is successful.

To validate the update, use the following steps:

1. All pods that are deployed should be in the running state before and after an update.

   ```
   kubectl get pods -n cn
   ```

2. Helm charts should reflect charts from the appropriate build.

   To check the helm charts currently deployed, use the following command in the cnSGW-C Ops Center.

   ```
   show helm charts
   show running-config helm repository
   ```

3. Check subscriber, session, or peer information for retention validation, using the following configuration:

   ```
   show subscriber namespace sgw count all
   show peers all
   ```

Rolling Upgrade Optimization

Table 7: Feature History

Feature Name	Release Information	Description
Rolling Upgrade Optimization for Protocol PFCP Pods	2024.04.0	For the protocol (PFCP) pods, Converged Core Gateway introduces a session-level response messages cache at the service pod for handling the retransmitted requests. This optimization helps in reduced session and Call Events Per Second (CEPS) loss during the upgrade procedure.

Feature Name	Release Information	Description
Rolling Upgrade Optimization for Endpoint Pods	2024.03.0	Converged Core Gateway provides the rolling upgrade optimization support for the following pods: • Radius endpoint pod • Diameter endpoint pod • UDP proxy pod • GTPP endpoint pod This optimization is an enhancement to the existing optimization. This feature uses the existing CLI command **supported-features [app-rx-retx-cache | app-tx-retx | rolling-upgrade-all | rolling-upgrade-enhancement-infra]** in the converged core profile.
Rolling Upgrade Optimization	2024.02.0	Converged Core Gateway provides the following support: • Retry mechanism at service and protocol pods during upgrades • Configuration-based rolling upgrade enhancements This optimization helps in reduced session and Call Events Per Second (CEPS) loss during the upgrade procedure. The configurable rolling upgrade enhancements enable smooth rollout of the changes. This feature introduces the new CLI command **supported-features [app-rx-retx-cache | app-tx-retx | rolling-upgrade-all | rolling-upgrade-enhancement-infra]** in the converged core profile. **Default Setting:** Disabled – Configuration Required

Converged Core Gateway (CCG) software version 2024.02.0 and higher support rolling upgrade with additional optimizations. Rolling upgrade lets you perform graceful upgrade of all pods with minimal impact on sessions

and CEPS. The additional optimizations provide an operational flexibility by allowing you an option to enable or disable rolling upgrade features through the CLI command for the upgrade process, as required.

This feature supports the following application-level enhancements:

- Retry mechanisms at protocol pods during service pods upgrade.
- Handling of transient sessions or transactions at service pods and protocol pods during their upgrades.
- Handling of topology and IPC mechanism changes to detect pods that are restarting or inactive. For inactive pods, the retry option is attempted toward other instances of pods.

You can configure the rolling upgrade enhancements through the **supported-features [app-rx-retx-cache | app-tx-retx | rolling-upgrade-all | rolling-upgrade-enhancement-infra]** CLI command.



Important

- It is recommended that you do not enable or disable the rolling upgrade features at run time to prevent an impact on the existing sessions.
- It is highly recommended that you use only the **rolling-upgrade-all** option as all the other command options are available only for debugging purpose.

Feature Description

Converged Core Gateway (CCG) software version 2024.02.0 and higher supports rolling upgrade with additional optimizations. Rolling upgrade lets you perform graceful upgrade of all pods with minimal impact on sessions and CEPS.

This feature supports the following application-level enhancements:

- Retry mechanisms at protocol pods during service pods upgrade.
- Handling of transient sessions or transactions at service pods and protocol pods during their upgrades.
- Handling of topology and IPC mechanism changes to detect pods that are restarting or inactive. For inactive pods, the retry option is attempted toward other instances of pods.

You can configure the rolling upgrade enhancements through the **supported-features [app-rx-retx-cache | app-tx-retx | rolling-upgrade-all | rolling-upgrade-enhancement-infra]** CLI command.



Important

- It is recommended that you do not enable or disable the rolling upgrade features at run time to prevent an impact on the existing sessions.
- It is highly recommended that you use only the **rolling-upgrade-all** option as all the other command options are available only for debugging purpose.

How Rolling Upgrade Optimization Works

This section describes upgrades of various pods and the rolling upgrade procedure.

Pod Upgrades

Service Pod

Peer pods are made aware of the upgrade or restart of a particular pod so that the transactions from/to that pod are handled gracefully.

The following section describes the handling of incoming and outgoing messages on a service pod:

Incoming Messages

- During an upgrade, the state of a service pod is communicated to other pods through the topology update. If no affinity exists for the session, the pod is not selected for forwarding new messages.
- For a session, if a service pod doesn't have a context, which is the first message for a user after a cache timeout, an Application Stop message is communicated to the protocol pod. This pod redirects messages to other service pod instances.
- If a service pod has a context for the session with no pending procedures or transactions for the session or user, an Application Stop message is communicated to the protocol pod. This pod redirects messages to other service pod instances. However, before communication to the protocol pod, the forceful Sync of Session State is done toward CDL. In addition, the affinity entry is removed for the session or user.

Outgoing Messages

- After completion of a call flow or a procedure for a session or user, if a service pod receives an upgrade or restart indication, then synchronization of the session or PDU state with CDL is performed. In addition, the affinity entry is removed for the service or user to disallow the triggering of further messages toward the service pod instance.
- For the existing call flows, which are in progress, the transactions are handled on the best-effort basis for call completion.

Protocol Pod

This section describes the handling of messages on the REST endpoint, GTPC endpoint, and protocol (PFCP) endpoint pods.

REST Endpoint Pod

The following section describes the handling of incoming and outgoing messages on the REST endpoint pod.

Incoming Messages

- For new TCP connections the ingress K8 service doesn't select a specific REST endpoint pod during an upgrade or restart. These requests are forwarded to other instances.
- After receiving an upgrade or restart indication, a GOAWAY frame is sent on the existing connections. By sending this frame, the new messages are sent on a new connection from the peer node.

Outgoing Messages

- After receiving the outgoing request messages from service pods, the Application Stop indication is communicated back to the service pod. With this communication, a service pod can select another REST endpoint instance to trigger the messages.

- For outgoing responses for the existing transient messages, the best effort is made to complete the transactions.

GTPC Endpoint Pod

Incoming Messages

Session level response messages cache is added at service pod to support handling of incoming request messages during GTPC endpoint pod restart. Service pods store response messages buffers based on the sequence number, source IP address, source port, and request message type.

- If there is a response message loss on wire due to GTPC endpoint pod restart, then peer retries the request message. This message is responded using a session level response message cache.
- Even if a response message isn't generated at a service pod, the message is detected as retransmission at the service and handled accordingly.

Outgoing Messages

The outgoing request messages during GTPC endpoint pod restart are handled in the following way:

- The service pod sets the request timeout interval and the number of retransmissions while doing BGIPC. The timeout interval and the number of transmissions are based on the N3 T3 defined for S5, S11, and S5E interfaces.
- Instead of the GTPC endpoint pod, the service pod sets the source port and sequence number.
- This mechanism helps in retransmission of the outgoing message when a response message is lost during pod restart. The service pod generates the sequence number and port. In this case, there's no ambiguity of messages on the wire for the peer also to detect the message as retransmission.

Node Manager Pod

One instance of a node manager pod is available to serve the calls during upgrades. The readiness probe and the timer are also configured for the upgrade scenario. In case a pod is inactive, the service pods retry the other node manager pod instances.

CDL Pod

CDL pods have multiple replicas for the service continuity during the upgrade process. Multiple connection streams are available towards CDL endpoints to minimize failures during upgrade and restart processes. In case of errors for these processes, the retry mechanism is also implemented towards another CDL endpoint connection.

Upgrading Software to Version with Rolling Upgrade Optimization Support

This section describes how to perform the rolling upgrade and to enable the rolling upgrade enhancements.

Important Review these important guidelines associated with the rolling upgrade procedure.

- The rolling upgrade optimization feature should only be used when you are upgrading from Release 2024.02.0 or later.

  If you are upgrading to Release 2024.02.0 or later from a version that is prior to Release 2024.02.0, perform the shut-start upgrade first.

- After the upgrade, make sure you enable the rolling upgrade enhancements using CLI command. Then, the subsequent rolling upgrades to future releases will include the available optimizations.

Rolling Upgrade Considerations

Both the racks (Rack 1 and Rack 2) of 5GSA GW are in a sunny day scenario and are on a version that is prior to Release 2024.02.0.

To perform the rolling strategy, follow these steps:

1. Move Rack 1 to a rainy day scenario and keep Rack 2 active for both the GR instances.
2. Move the GR instances on Rack 1 to Standby_ERROR.
3. Shutdown Rack 1.
4. Perform cluster sync, through sync-phase ops-center, for Rack 1 to move to Release 2024.02.
5. Apply the recommended configurations to enable the rolling upgrade enhancements.

   The recommended configuration for rolling upgrade is as follows:

   ```
   config
     profile converged-core converged_core_profile_name
     supported-features [ rolling-upgrade-all ]
     end
   ```

6. Start Rack 1.
7. Wait for 30 minutes for completion of CDL reconciliation.
8. Switch the GR instances to Primary to make Rack 1 active.
9. Shutdown Rack 2.
10. Continue with Steps 4–6 for Rack 2.
11. Make the Rack 1 and Rack 2 configurations for a sunny day scenario.

Limitations

This feature has the following limitations:

- During a rolling upgrade, service pods restart one at a time. This upgrade leads to a skewed redistribution of sessions. The service pod that restarts first has the higher number of sessions. Similarly, the service pod that restarted last has the least number of sessions. Such redistribution of sessions can lead to a temporary spike in the memory requirement for some service pods. The system works as expected after the sessions are removed from the local cache of a service pod.

- Ongoing procedures in service pods continue during the rolling upgrade. However, the best effort mechanism is implemented for their successful completion.

Configuring the Supported Features for Rolling Upgrade

To enable the supported features for a rolling upgrade, use the following sample configuration:

```
config
  profile converged-core cc_profile_name
     supported-features [ app-rx-retx-cache | app-tx-retx |
rolling-upgrade-all | rolling-upgrade-enhancement-infra ]
     end
```

NOTES:

- **profile converged-core** *cc_profile_name*: Specify the name of the converged core profile. This keyword allows you to enter the converged core profile configuration mode.
- **supported-features [app-rx-retx-cache | app-tx-retx | rolling-upgrade-all | rolling-upgrade-enhancement-infra]**: Specify one of the following options to enable the supported features for the rolling upgrade.
  - **app-rx-retx-cache**: Enable retransmission cache for inbound messages at application.
  - **app-tx-retx**: Enable retransmission for outbound messages at application.
  - **rolling-upgrade-all**: Enable all the rolling upgrade features that are available through **rolling-upgrade-enhancement-infra**, **app-rx-retx-cache**, and **app-tx-retxrolling** keyword options. By default, the rolling upgrade features are disabled.

    rolling-upgrade-all is the only recommended option.
  - **rolling-upgrade-enhancement-infra**: Enable infra-level features.



Important

- It is recommended that you do not enable or disable the rolling upgrade features at run time to prevent an impact on the existing sessions.
- It is highly recommended that you use only the **rolling-upgrade-all** option as all the other command options are available only for debugging purpose.

Verifying Rolling Upgrade Optimization

Use the **show running-config profile amf-services** command to verify the supported features for a rolling upgrade.

The following is an example output of the **show running-config amf-services** command.

```
show running-config profile amf-services
amf-services am1
   supported-features [ rolling-upgrade-all ]
exit
```

OAM Support

Bulk Statistics

The following statistics are supported for the rolling upgrade optimization feature.

IPC retry statistics:

The "ipc_request_total" statistics is updated with an additional label "retry_cause" for the cause of a retry attempt.

CDL statistics:

The following statistics are added for the CDL operations:

- cdl_request_total
- cdl_response_total
- cdl_request_seconds_total
- cdl_request_duration_histogram_total

These CDL statistics are updated with the following filters:

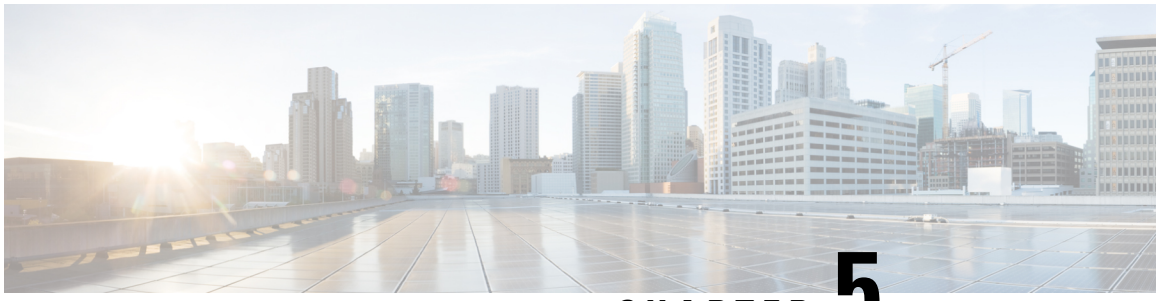
- **retry**: Used to view the retry messages
- **method_name**: Used to view the CDL force sync update



Note In the previous releases, CDL statistics were visible through RPC statistics with the filter **rpc_name** as STREAM_SESSION_DB. From Release 2024.02 onwards, CDL statistics are available only using the preceding CDL statistics.

Application stop counter:

The "application_stop_action" statistics is added to view actions on App-infra. Some examples of these actions are session cache removal and affinity removal.



CHAPTER 5

Pods and Services Reference

- Feature Summary and Revision History, on page 65
- Feature Description, on page 65
- Associating Pods to the Nodes, on page 73
- Viewing the Pod Details and Status, on page 74
- Runtime pod scales for server additions, on page 75

Feature Summary and Revision History

Summary Data

Table 8: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Revision Details	Release
First introduced	2020.07

Feature Description

cnSGW-C is built on the Kubernetes cluster strategy, adopting the native concepts of containerization, high availability, scalability, modularity, and ease of deployment. cnSGW-C uses the components, such as pods and services offered by Kubernetes.

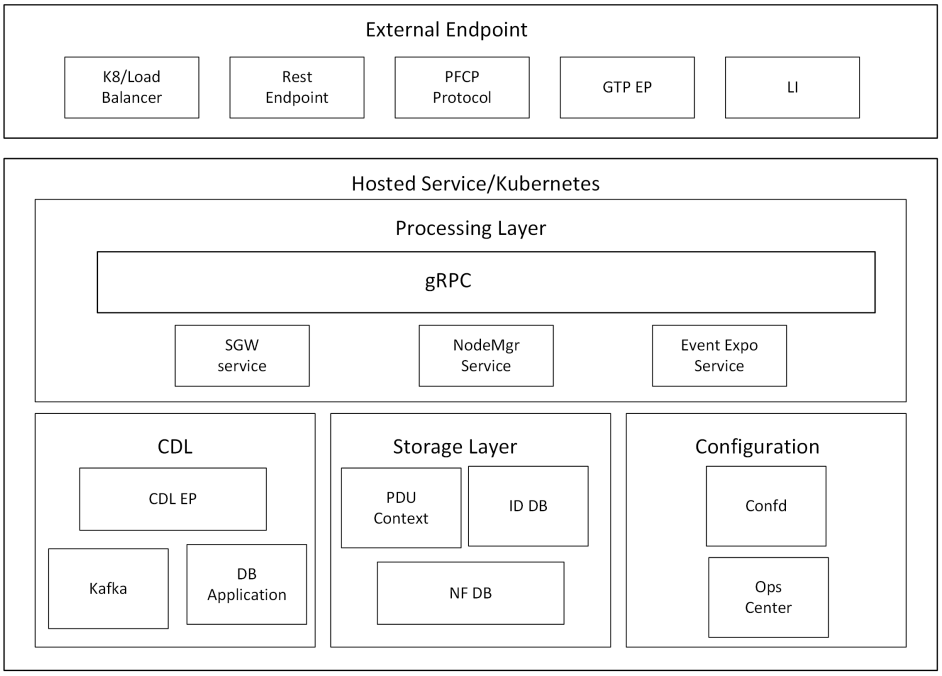
Depending on your deployment environment, the cnSGW-C deploys the pods on the configured virtual machines(VM) that you have configured. Pods operate through the services that are responsible for the intrapod communications. If the machine hosting the pods fail or experiences network disruption, the pods are terminated or deleted. However, this situation is transient and k8s, create new pods to replace the invalid pods.

The following workflow provides high-level information about:

- Host machines
- Associated pods and services
- Interaction among pods

The representation might defer based on your deployment infrastructure.

Figure 14: Communication Workflow of Pods



Kubernetes deployment includes the kubectl command-line tool to manage the Kubernetes resources in the cluster. You can manage the pods, nodes, and services.

For generic information on the Kubernetes concepts, see the Kubernetes documentation.

Pods

A pod is a process that runs on Kubernetes cluster. Pod encapsulates a granular unit known as a container. A pod can contains one or more containers.

Kubernetes deploys one or multiple pods on a single node which can be a physical or a virtual machine. Each pod has a discrete identity with an internal IP address and port number. The containers within the pod shares the storage and network resources.

The following tables list the cnSGW-C and Common Execution Environment (CEE) pod names and the hosts on which they are deployed depending on the labels that you assign. See the following table for information on how to assign the labels.

Table 9: cnSGW-C Pods

Pod Name	Description	Host Name
api-sgw-ops-center	Functions as *confD* API pod for the cnSGW-C Ops Center.	OAM
base-entitlement-sgw	Operates to support smart licensing feature.	OAM **Note** Currently not suppor
bgpspeaker	Operates to support dynamic routing for L3 route management and BFD monitoring.	Protocol
cache-pod	Operates to support cache system information that is used by other pods as applicable.	Protocol
cdl-ep-session-c1	Provides an interface to the CDL.	Session
cdl-index-session-c1	Preserves the mapping of keys to the session pods.	Session
cdl-slot-session-c1	Operates as the CDL session pod to store the session data.	Session
documentation	Contains the documentation.	OAM
etcd-sgw-etcd-cluster	Hosts the etcd for the cnSGW-C OAM application to store information such as pod instances, leader information, endpoints.	OAM
georeplication	Operates to support cache, ETCD replication across sites, and site role management.	Protocol
grafana-dashboard-app-infra	Contains the default dashboard of app-infra metrics in Grafana.	OAM
grafana-dashboard-cdl	Contains the default dashboard of CDL metrics in Grafana.	OAM
grafana-dashboard-sgw	Contains the default dashboard of cnSGW-C service metrics in Grafana.	OAM
gtpc-ep-n0	Operates as GTPC endpoint of cnSGW-C.	Protocol
kafka	Hosts the Kafka details for the CDL replication.	Protocol
li-ep-n0	Operates as Lawful Intercept endpoint of cnSGW-C.	Protocol
oam-pod	Operates as the pod to facilitate Ops Center actions, such as show commands, configuration commands, monitor protocol monitor subscriber.	OAM

Pod Name	Description	Host Name
ops-center-sgw-ops-center	Acts as the cnSGW-C Ops Center.	OAM
smart-agent-sgw-ops-center	Operates as the utility pod for the cnSGW-C Ops Center.	OAM
nodemgr-n0	Performs node level interactions, such as Sxa link establishment and management (heartbeat). It generates unique identifiers, such as UE IP address and SEID.	Service
protocol-n0	Operates as encoder and decoder of application protocols (PFCP, whose underlying transport protocol is UDP).	Protocol
rest-ep-n0	cnSGW-C uses REST-EP as Notification client.	Protocol
service-n0	Contains main business logic of cnSGW-C.	Service
udp-proxy	Operates as proxy for all UDP messages. Owns UDP client and server functionalities.	Protocol
swift-sgw-ops-center	Operates as the utility pod for the cnSGW-C Ops Center.	OAM
zookeeper	Assists Kafka for topology management.	OAM

CEE Pods

For details, see the "CEE pods" topic from the UCC Common Execution Environment - Configuration and Administration Guide.

Service pod

The Service Pod performs subscriber session management and control-plane processing for the Converged Core deployment. In a ULB-enabled deployment, the Service Pod also processes GTPC and PFCP signaling received from the Unified Load Balancer (ULB). The ULB forwards signaling traffic directly to the Service Pod while maintaining session affinity throughout the subscriber session lifecycle. Integrating GTPC and PFCP signaling processing within the Service Pod reduces inter-pod signaling and simplifies the signaling path.

UDP Proxy Pod

Feature Description

The cnSGW-C has UDP interfaces towards the UP (Sxa), MME (S11), and PGW (S5 or S8). With the help of the protocol layer pods, the messages are encoded, decoded, and exchanged on these UDP interfaces.

For achieving the functionalities mentioned on the 3GPP specifications:

- It is mandatory for the protocol layer pods to receive the original source and destination IP address and port number. But the original IP and UDP header is not preserved when the incoming packets arrive at the UDP service in the Kubernetes (K8s) cluster.

- Similarly, for the outgoing messages, the source IP set to the external IP address of the UDP service (published to the peer node) is mandatory. But the source IP is selected as per the egress interface when different instances of protocol layer pods send outgoing messages from different nodes of the K8s cluster.

The protocol layer pod spawns on the node, which has the physical interface configured with the external IP address to achieve the conditions mentioned earlier. However, spawning the protocol layer pods has the following consequences:

- It is not possible to achieve the node level HA (High Availability) as the protocol pods are spawned on the same node of the K8s cluster. Any failure to that node may result in loss of service.
- The protocol pods must include their own UDP client and server functionalities. In addition, each protocol layer pod may require labeling of the K8s nodes with the affinity rules. This restricts the scaling requirements of the protocol layer pods.

The cnSGW-C addresses these issues with the introduction of a new K8s pod called udp-proxy. The primary objectives of this pod are:

- The udp-proxy pod acts as a proxy for all kinds of UDP messages. It also owns the UDP client and server functionalities.
- The protocol pods perform the individual protocol (PFCP, GTP, Radius) encoding and decoding, and provide the UDP payload to the udp-proxy pod. The udp-proxy pod sends the UDP payload out after it receives the payload from the protocol pods.
- The udp-proxy pod opens the UDP sockets on a virtual IP (VIP) instead of a physical IP. This ensures that the udp-proxy pod does not have any strict affinity to a specific K8s node (VM), thus enabling node level HA for the UDP proxy.

Note One instance of the udp-proxy pod is spawned by default in all the worker nodes in the K8s cluster.

The UDP proxy for cnSGW-C feature has functional relationship with the Virtual IP Address feature.

Architecture

The udp-proxy pod is placed in the worker nodes in the K8s cluster.

1. Each of the K8s worker node contains one instance of the udp-proxy pod. However, only one of the K8s worker node owns the virtual IP at any time. The worker node that owns the virtual IP remains in the active mode while all the other worker nodes remain in the standby mode.
2. The active udp-proxy pod binds to the virtual IP and the designated ports for listening to the UDP messages from the peer nodes (UPF and SGW).
3. The UDP payload received from the peer nodes are forwarded to one instance of the protocol, gtp-ep, or radius-ep pods. The payload is forwarded either on the same node or different node for further processing.
4. The response message from the protocol, gtp-ep, or radius-ep pods is forwarded back to the active instance of the udp-proxy pod. The udp-proxy pod sends the response message back to the corresponding peer nodes.
5. The cnSGW-C-initiated messages are encoded at the protocol, gtp-ep, or radius-ep pods. In addition, the UDP payload is sent to the udp-proxy pod. Eventually, the udp-proxy pod comprises of the complete IP

payload and sends the message to the peer. When the response from the peer is received, the UDP payload is sent back to the same protocol pod from which the message originated.

Protocol Pod Selection for Peer-Initiated Messages

When the udp-proxy pod receives the peer node (for instance UPF) initiated messages, it is load-balanced across the protocol instances to select any instance of the protocol pod. An entry of this instance number is stored along with the source IP and source port number of the peer node. This ensures that the messages form the same source IP and source port are sent to the same instance that was selected earlier.

High Availability for the UDP Proxy

The UDP proxy's HA model is based on the keepalived virtual IP concepts. A VIP is designated to the N4 interface during the deployment. Also, a keepalived instance manages the VIP and ensures that the IP address of the VIP is created as the secondary address of an interface in one of the worker nodes of the K8s cluster.

The udp-proxy instance on this worker node binds to the VIP and assumes the role of the active udp-proxy pod. All udp-proxy instances in the other worker nodes remain in the standby mode.

Services

The cnSGW-C configuration is composed of several microservices that run on a set of discrete pods. These Microservices are deployed during the cnSGW-C deployment. cnSGW-C uses these services to enable communication between the pods. When interacting with another pod, the service identifies the pod's IP address to initiate the transaction and acts as an endpoint for the pod.

The following table describes the cnSGW-C services and the pod on which they run.

Table 10: cnSGW-C Services and Pods

Service Name	Pod Name	Description
base-entitlement-sgw	base-entitlement-sgw	Operates to support sma
bgpspeaker-pod	bgpspeaker	Operates to support dyn route management and
datastore-ep-session	cdl-ep-session-c1	Responsible for the CDL
datastore-notification-ep	smf-rest-ep	Responsible for sending the CDL to the *sgw-servi* **Note** cnSGW-C uses REST-E client.
datastore-tls-ep-session	cdl-ep-session-c1	Responsible for the secu
documentation	documentation	Responsible for the cnS
etcd	etcd-sgw-etcd-cluster-0, etcd-sgw-etcd-cluster-1, etcd-sgw-etcd-cluster-2	Responsible for pod dis namespace.

Service Name	Pod Name	Description
etcd-sgw-etcd-cluster-0	etcd-sgw-etcd-cluster-0	Responsible for sync the *etcd* cluster.
etcd-sgw-etcd-cluster-1	etcd-sgw-etcd-cluster-1	Responsible for sync the *etcd* cluster.
etcd-sgw-etcd-cluster-2	etcd-sgw-etcd-cluster-2	Responsible for sync the *etcd* cluster.
grafana-dashboard-app-infra	grafana-dashboard-app-infra	Responsible for the app-infra metrics in
grafana-dashboard-cdl	grafana-dashboard-cdl	Responsible for the metrics in Grafana.
grafana-dashboard-sgw	grafana-dashboard-sgw	Responsible for the cnSGW-C service m
gtpc-ep	gtpc-ep-n0	Responsible for inter GTP-C pod.
helm-api-sgw-ops-center	api-sgw-ops-center	Manages the Ops Ce
kafka	kafka	Processes the Kafka
li-ep	li-ep-n0	Responsible for law
local-ldap-proxy-sgw-ops-center	ops-center-sgw-ops-center	Responsible for leve credentials by other
oam-pod	oam-pod	Responsible to facili Ops Center.
ops-center-sgw-ops-center	ops-center-sgw-ops-center	Manages the cnSGW
ops-center-sgw-ops-center-expose-cli	ops-center-sgw-ops-center	To access cnSGW-C IP address.
smart-agent-sgw-ops-center	smart-agent-sgw-ops-center	Responsible for the c
smf-nodemgr	smf-nodemgr	Responsible for inter *smf-nodemgr* pod.
smf-protocol	smf-protocol	Responsible for inter smf-protocol pod.
sgw-service	sgw-service	Responsible for inter cnSGW-C service p
swift-sgw-ops-center	swift	Operates as the utili Ops Center.
zookeeper	zookeeper	Assists Kafka for to
zookeeper-service	zookeeper	Assists Kafka for to

Open Ports and Services

The cnSGW-C uses different ports for communication. The following table describes the default open ports and the associated services.

Table 11: Open Ports and Services

Port	Type	Service	Usage
22	tcp	SSH	SMI uses TCP port to communicate with the virtual machines.
53	tcp	domain	DNS port.
80	tcp	HTTP	SMI uses TCP port for providing Web access to CLI, Documentation, and TAC.
111	tcp	rpcbind	Open Network Computing Remote Procedure Call.
179	tcp	bgp	Border Gateway Protocol (BGP)
443	tcp	SSL/HTTP	SMI uses TCP port for providing Web access to CLI, Documentation, and TAC.
2379	tcp	etcd-client	CoreOS etcd client communication.
6443	tcp	http	SMI uses port to communicate with the Kubernetes API server.
7472	tcp	unknown	speaker, used by Grafana.
8083	tcp	us-srv	Kafka connects REST interface.
8850	tcp	unknown	udp-proxy
8879	tcp	unknown	udp-proxy
9100	tcp	jetdirect	SMI uses TCP port to communicate with the Node Exporter. Node Exporter is a Prometheus exporter for hardware and OS metrics with pluggable metric collectors. It allows you to measure various machine resources, such as memory, disk, and CPU utilization.
10250	tcp	SSL/HTTP	SMI uses TCP port to communicate with Kubelet. Kubelet is the lowest level component in Kubernetes. It is responsible for what is running on an individual machine. It is a process watcher or supervisor focused on active container. It ensures the specified containers are up and running.

Port	Type	Service	Usage
10251	tcp	-	SMI uses TCP port to interact with the Kube scheduler. Kube scheduler is the default scheduler for Kubernetes and runs as part of the control plane. A scheduler watches for newly created pods that have no node assigned. For every pod that the scheduler discovers, the scheduler becomes responsible for finding the best node for that pod to run on.
10252	tcp	apollo-relay	SMI uses this TCP port to interact with the Kube controller. The Kubernetes controller manager is a daemon that embeds the core control loops shipped with Kubernetes. The controller is a control loop that watches the shared state of the cluster through the API server and makes changes to move the current state to the desired state.
10256	-	HTTP	SMI uses TCP port to interact with the Kube proxy. Kube proxy is a network proxy that runs on each node in your cluster. Kube proxy maintains network rules on nodes. These network rules allow network communication to your pods from network sessions inside or outside of your cluster.
50051	tcp	unknown	gRPC service listen port.
53	udp	domain ISC BIND (Fake version: 9.11.3- 1ubuntu1.9- Ubuntu)	DNS port
111	udp	rpcbin	Open Network Computing Remote Procedure Call
2123	udp	gtpc	GTP control
8805	udp	pfcp	Packet Forwarding Control Protocol (PFCP)

Associating Pods to the Nodes

This section describes how to associate a pod to the node.

After configuring a cluster, you can associate the pods to the nodes through labels. This association enables the pods to get deployed on the appropriate node, based on the key-value pair.

Labels are required for the pods to identify the nodes where they must be deployed and to run the services. For example, when you configure the protocol-layer label with the required key-value pair, the pods are deployed on the nodes that match the key-value pair.

1. To associate pods to the nodes through the labels, use the following configuration:

```
config
 k8
   label
     cdl-layer
       key key_value
       value value
     oam-layer
       key key_value
       value value
     protocol-layer
       key key_value
       value value
     service-layer
       key key_value
       value value
       end
```

NOTES:

- If you don't configure the labels, cnSGW-C assumes the labels with the default key-value pair.
  - **label { cdl-layer { key** *key_value* **| value** *value* **}**—Configures the key value pair for CDL.
  - **oam-layer { key** *key_value* **| value** *value* **}**—Configures the key value pair for OAM layer.
  - **protocol-layer { key** *key_value* **| value** *value* **}**—Configures the key value pair for protocol layer.
  - **service-layer { key** *key_value* **| value** *value* **}**—Configures the key value pair for the service layer.

Viewing the Pod Details and Status

If the service requires additional pods, cnSGW-C creates and deploys the pods. You can view the list of available pods in your deployment through the cnSGW-C Ops Center.

You can run the kubectl command from the master node to manage the Kubernetes resources.

Pod Details

1. To view the comprehensive pod details, use the following command.

   ```
   kubectl get pods -n sgw pod_name -o yaml
   ```

The output of this command provides the pod details in YAML format with the following information:

- The IP address of the host where the pod is deployed.
- The service and the application that is running on the pod.
- The ID and the name of the container within the pod.
- The IP address of the pod.
- The present state and phase of the pod.

- The start time from which pod is in the present state.

Use the following command to view the summary of the pod details.

```
kubectl get pods -n sgw_namespace -o wide
```

States

The following table describes the state of a pod.

Table 12: Pod States

State	Description
Running	The pod is healthy and deployed on a node. It contains one or more containers.
Pending	The application is in the process of creating the container images for the pod.
Succeeded	Indicates that all the containers in the pod are successfully terminated. These pods cann't be restarted.
Failed	One ore more containers in the pod have failed the termination process. The failure occurred as the container either exited with non-zero status or the system terminated the container.

Runtime pod scales for server additions

Table 13: Feature History

Feature Name	Release Information	Description
Scaling Converged Core Cluster with Node Labeling and Required Affinity	2025.03.0	This feature enables the dynamic addition of servers to a running cluster without disrupting existing deployments. It allows seamless scalability to meet growing demands and minimizes downtime, enhancing operational continuity.

The runtime pod scales feature manages scaling efficiently when a new server is added.

The key features and capabilities include:

- Enables dynamic recalibration of resources during new server addition.
- Utilizes new resources at runtime in a standalone converged core cluster.

- Supports Kubernetes version 1.33.

Key advantages of the pod scaling

The pod scaling feature has key advantages:

- Dynamic server addition.
- Usage of internal Kubernetes labels and affinity rules for workload management.
- Efficient load distribution.

Addition and scaling of runtime servers in converged core process

When a new server is added to the Kubernetes cluster using the cluster manager, ensure that the replica count for the “smf service” and “sgw-service” endpoints is increased, resulting in application pods being scheduled on the new node.

Complete these actions:

- Updating the Helm charts for both smf-service and sgw-service pods to use the required affinity rules.
- Labeling the new node.
- Increasing replica counts in the endpoint configuration.
- Removing the label and performing any other necessary cleanup

Prerequisite

The prerequisites for this feature are:

- Decommission the CF or SF Node from VPC-DI.
- Clean up the Virtual Drive of this CF or SF node.
- Connect the node towards the 5G cluster by physically attaching PCIe slots.
- Ensure the SMI CNDP version is the July 2025 release version.
- Ensure that the OpsCenter is set to the "large-all" deployment model before proceeding.
- Run Full Cluster Sync to get this node added to the cluster.
- Set the node count for the endpoint service and sgw-service to 0 or 1.

Configure new deployment

supports a new deployment model "large-all". By default, "large" model is the default value used on existing deployments.

Use these Day 1 configuration steps for new deployment.

1. Enter the configuration mode.

   ```
   configure
   ```

2. Specify the deployment model using the **deployment model** *deployment-model* command.

   Example

   ```
   [dev-perf/smf] smf# config
   [dev-perf/smf] smf(config)# deployment model large-all
   [dev-perf/smf] smf(config)# commit
   ```

3. Enter **commit** to commit the changes.

Pod scaling processes

The process includes these steps:

1. Log into the Ops Center using the **cn-ops-center** command, to label the new node.

   Enter **cluster label node** *node_name* **pod-list** *pod-list* to label the node for specific pod scaling.

   Example

   ```
   cluster label node dev-perf-worker-4 pod-list {sgw-service|smf-service}
   ```

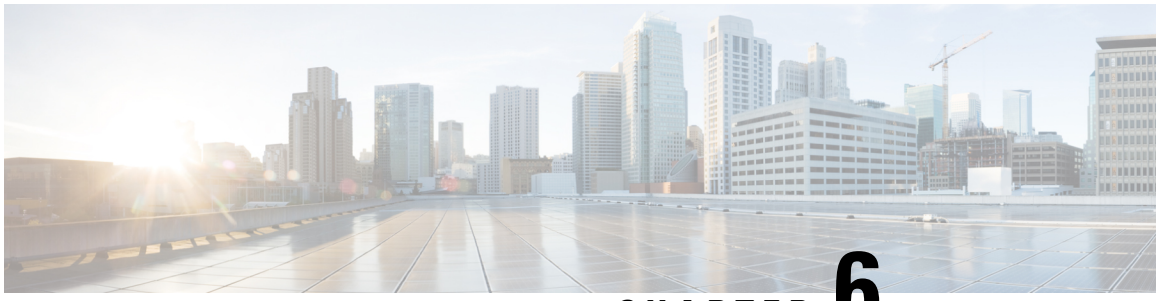
2. Enter into the configuration mode and increase the replica count under the endpoint service (for) or increase the replica count for sgw-service.
3. Wait for the system status to be 100%.
4. Clear the node label for specific pod scaling using the **cluster unlabel node** *node_name* **pod-list** *pod-list*

   example

   ```
   cluster unlabel node dev-perf-worker-4 pod-list {sgw-service|smf-service}
   ```

5. Verify the node labels status using the **show cluster label** command.

   ```
   dev-perf/smf] smf# show cluster label
   NAME               STATUS
   -------------------------------------------
   dev-perf-master-1
   dev-perf-master-2
   dev-perf-master-3
   dev-perf-worker-1
   dev-perf-worker-2
   dev-perf-worker-3
   dev-perf-worker-4  smf-service,sgw-service
   ```



CHAPTER 6

3GPP RAN/NAS Cause Codes Support

- Feature Summary and Revision History, on page 79
- Feature Description, on page 79
- How it Works, on page 81

Feature Summary and Revision History

Summary Data

Table 14: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 15: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

cnSGW-C supports RAN/NAS cause codes as defined in *3GPP TS 29.274*, *version 15.4.0*, *section 8.103*, *RAN/NAS Cause*.

cnSGW-C transparently transmits the RAN/NAS Release Cause IE provided by the MME to the PGW for further propagation towards the PCRF.

Note GTP-based S5/S8 and S11 are supported.

The following table lists the RAN/NAS Cause codes.

Table 16: RAN/NAS Cause Codes

Octets	Bits							
	8	7	6	5	4	3	2	1
1	Type = 172 (decimal)							
2–3	Length = n							
4	Spare				Instance			
5	Protocol Type				Cause Type			
6 to m	Cause Value							
(m+1) to (n+4)	One or more octets from these octets are present, only if explicitly specified							

The Protocol Type field is encoded with the specified values for the RAN/NAS Cause as follows:

Table 17: Protocol Type

Protocol Type	Values (Decimal)
S1AP Cause	1
EMM Cause	2
ESM Cause	3
<spare>	4–15

The Cause Value field (and the associated RAN cause subcategory) is transferred over the S1-AP interface. The field is encoded in one octet as a binary integer.

Table 18: Cause Type

Cause Type	Values (Decimal)
Radio Network Layer	0
Transport Layer	1
NAS	2
Protocol	3
Miscellaneous	4

Cause Type	Values (Decimal)
<spare>	5–15

For EMM and ESM Causes, the Cause Value field contains the cause value as specified in *3GPP TS 24.301*. If the Protocol is S1AP, the cause value contains the specified value as in *3GPP TS 36.413*.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for the RAN/NAS Cause Codes feature.

Create Bearer Procedure Call Flow

This section describes the create bearer procedure call flow.

Figure 15: Create Bearer Procedure Call Flow

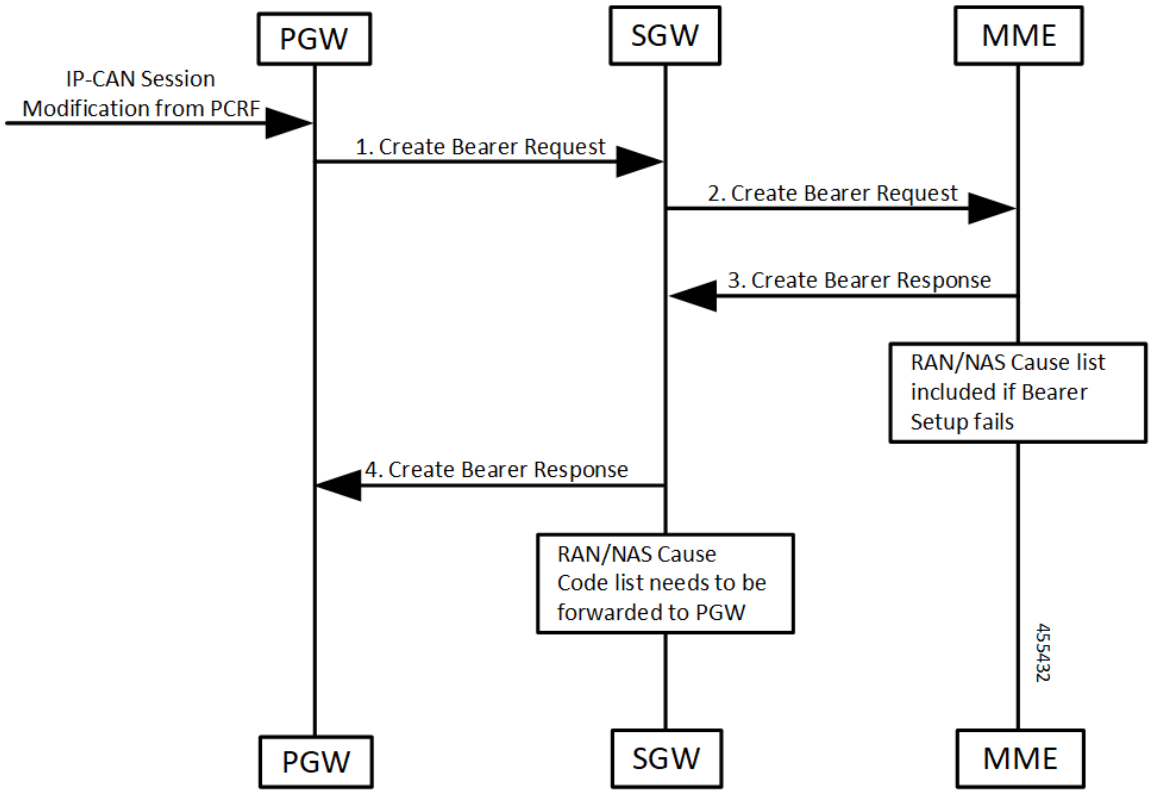


Table 19: Create Bearer Procedure Call Flow Description

Step	Description
1	PGW receives IP-CAN session modification request from PCRF. PGW creates the Create Bearer Request message and sends it to SGW (cnSGW-C).
2	SGW (cnSGW-C) forwards the Create Bearer Request message request to MME.
3	MME generates a Create Bearer Response message towards SGW (cnSGW-C). If bearer setup fails, then the RAN/Cause list included in the response.
4	SGW (cnSGW-C) forwards the Create Bearer Response message to PGW. It includes RAN/NAS Cause Code list.

Update Bearer Procedure Call Flow

This section describes the update bearer procedure call flow.

Figure 16: Update Bearer Procedure Call Flow

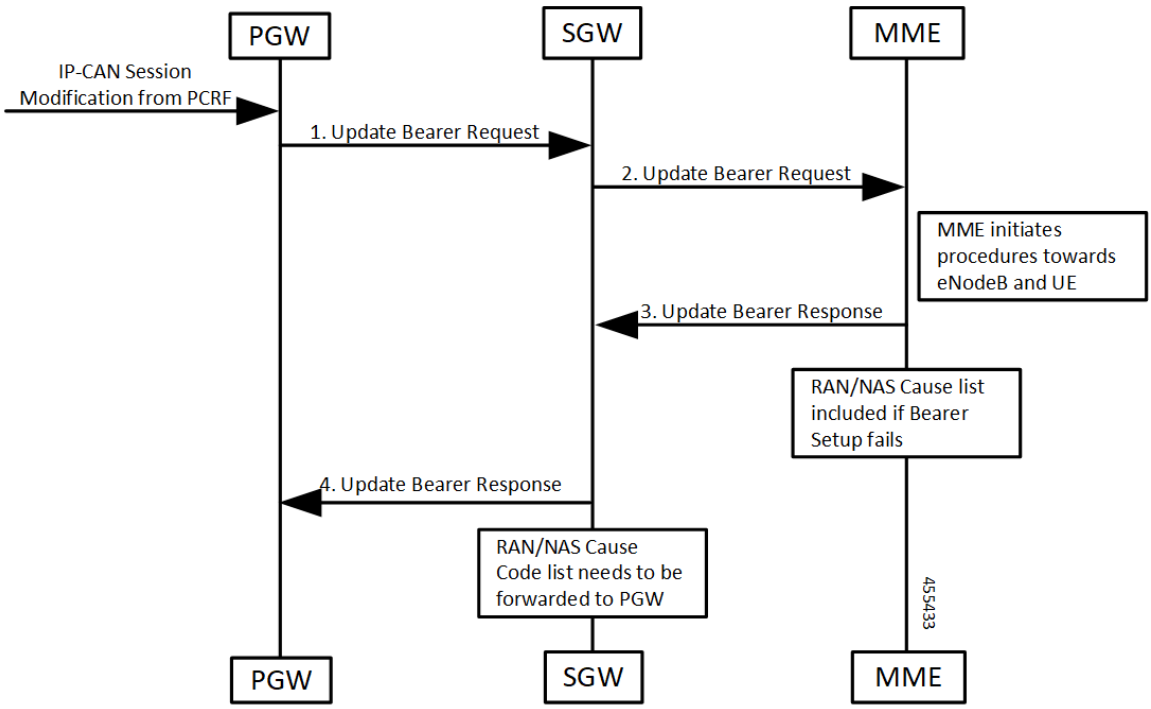


Table 20: Update Bearer Procedure Call Flow Description

Step	Description
1	PGW receives IP-CAN session modification request from PCRF. PGW creates the Update Bearer Request message to SGW (cnSGW-C).

Step	Description
2	SGW (cnSGW-C) forwards the Update Bearer Request message to MME.
3	MME generates an Update Bearer Response message towards SGW (cnSGW-C). If this bearer modification fails, then the RAN/NAS list included in the response.
4	SGW (cnSGW-C) forwards the Update Bearer Response message to PGW.

Delete Bearer Command Procedure Call Flow

This section describes the delete bearer command procedure call flow.

Figure 17: Delete Bearer Command Procedure Call Flow

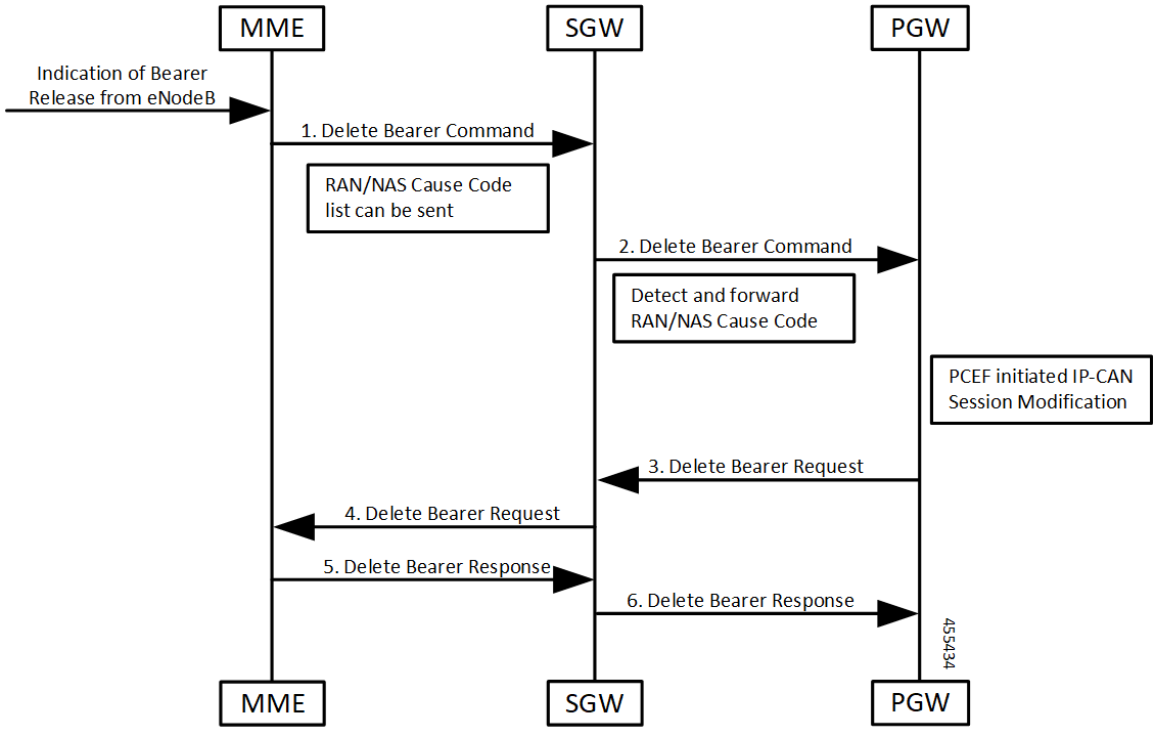


Table 21: Delete Bearer Command Procedure Call Flow Description

Step	Description
1	MME receives an indication of Bearer Release from eNodeB request. MME creates Delete Bearer Command message to SGW (cnSGW-C). It includes RAN/NAS cause code list.
2	SGW (cnSGW-C) forwards the Delete Bearer Command message request to PGW. It detects and forwards RAN/NAS cause code list.

Step	Description
3	PGW sends the Delete Bearer Request message to SGW (cnSGW-C). PGW receives IP-CAN session modification request from PCEF.
4	SGW (cnSGW-C) generates a Delete Bearer Request message towards MME.
5	MME generates a Delete Bearer Response message towards SGW (cnSGW-C).
6	SGW (cnSGW-C) further sends the Delete Bearer Response message to PGW.

Delete Session Procedure Call Flow

This section describes the delete session procedure call flow.

Figure 18: Delete Session Procedure Call Flow

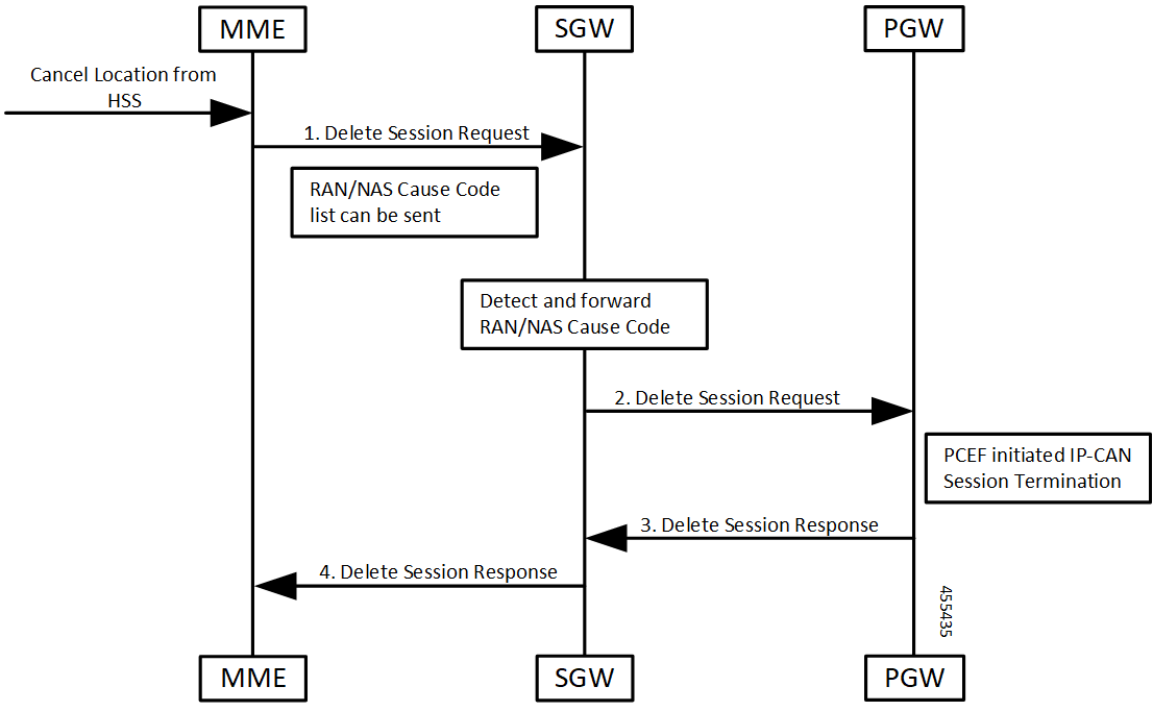
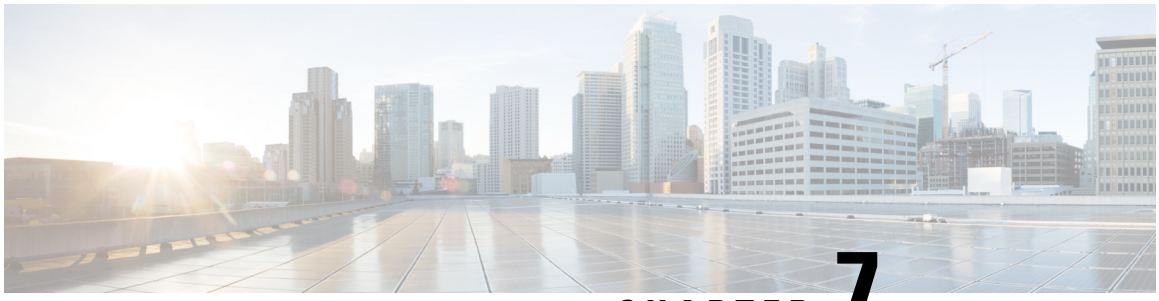


Table 22: Delete Session Procedure Call Flow Description

Step	Description
1	MME receives an indication of Cancel Location from HSS. MME creates Delete Session Request message to SGW (cnSGW-C). It includes RAN/NAS cause code list.
2	SGW (cnSGW-C) forwards the Delete Session Request message request to PGW. It detects and forwards RAN/NAS cause code list.

Step	Description
3	PGW sends the Delete Session Response message to SGW (cnSGW-C). PGW receives IP-CAN session modification request from PCEF.
4	SGW (cnSGW-C) generates a Delete Session Response message towards MME.



CHAPTER 7

Access Bearer Release Support

- Feature Summary and Revision History, on page 87
- Feature Description, on page 87
- How it Works, on page 88

Feature Summary and Revision History

Summary Data

Table 23: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 24: Revision History

Revision Details	Release
First introduced.	2020.03.0

Feature Description

cnSGW-C supports the handling of the Release Access Bearer (RAB) request procedure. It's a UE-level message. In multiple PDN scenarios, the MME sends only one RAB message, which applies to all the PDNs.

cnSGW-C brings all the bearers of all the PDNs to the IDLE state.

How it Works

This section describes how this feature works.

cnSGW-C sends the Sx Modification Request message per PDN to the corresponding User Plane. After receiving the Sx Modification response message from all user planes (for all PDNs), cnSGW-C sends the response message to MME.

cnSGW-C updates the state as IDLE for all the bearers in CDL.

Call Flows

This section describes the key call flow for the Access Bearer Release Support feature.

Release Access Bearer (Active to IDLE Transaction) Call Flow

This section describes the Release Access Bearer call flow.

Figure 19: Release Access Bearer (Active to IDLE Transaction) Call Flow

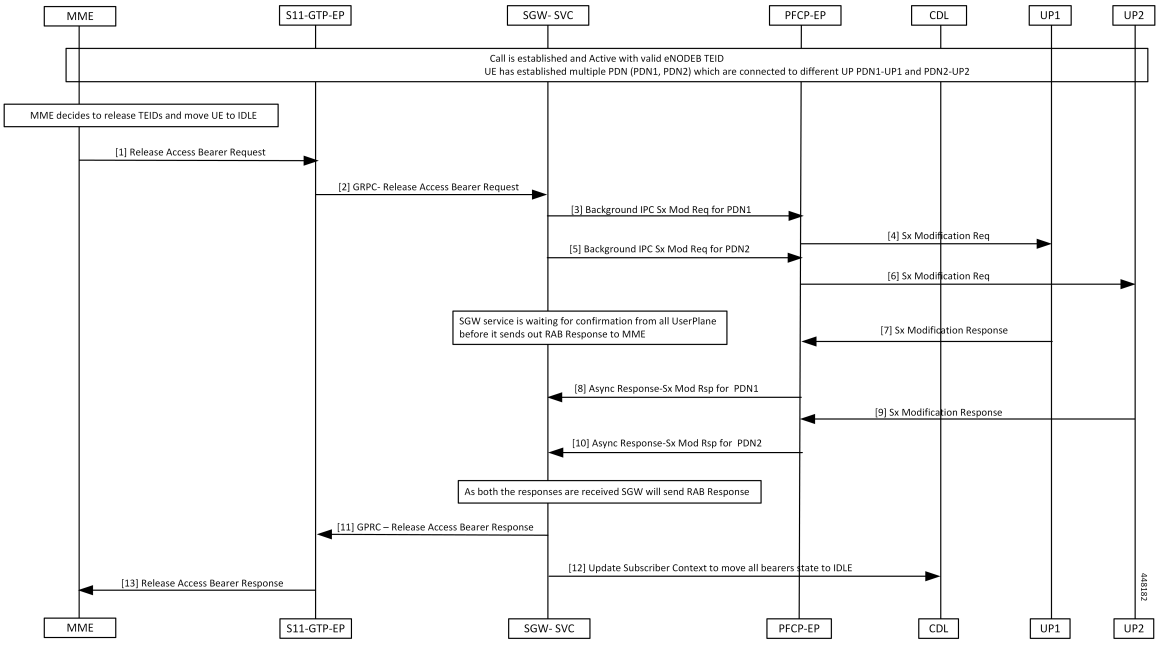
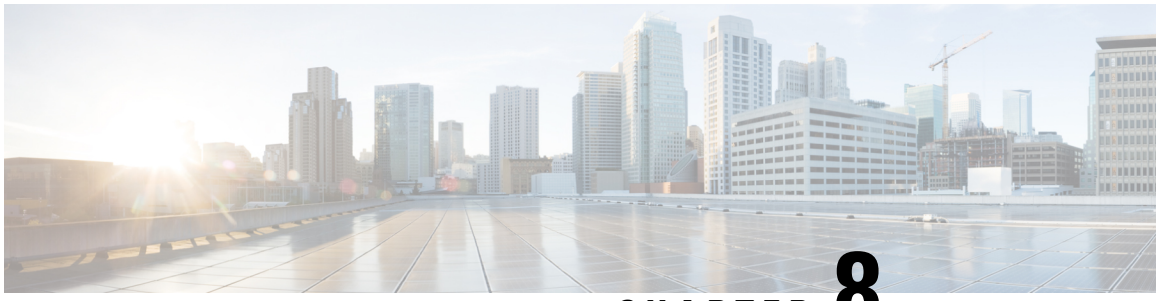


Table 25: Release Access Bearer (Active to IDLE Transaction) Call Flow Description

Step	Description
1	MME sends Release Access Bearer (RAB) request to S11-GTP-EP to release all S1-U bearers for the UE.

Step	Description
2	S11-GTP EP decodes the received UDP message and converts it into gRPC. The converted gRPC message then sent to the SGW-Service pod, using the TEID value, which can handle this UE session. SGW-Service pod performs the following activities: • Finds out Subscriber Context using local ingress TEID • Validates the RAB request content • Moves UE to the IDLE state • Builds the Sx Modify request message with the downlink apply action as DROP, to drop all downlink packets at SGW-U
3	SGW-Service pod sends the Sx Mod request message using the background IPC async call for PDN1 to PFCP-EP.
4	PFCP-EP forwards the Sx Modify Request (PDN1) message to UPF1 through the UDP proxy. UPF1 processes the Sx Modify Request (PDN1) message.
5	SGW-Service pod sends the Sx Modify Request message using the background IPC async call for PDN2. PFCP-EP forwards the Sx Modify Request (PDN2) message to UPF2 through the UDP proxy.
6	UPF2 processes the Sx Modify Request (PDN2) message.
7	UPF1 sends the Sx Modify response (PDN1) message to PFCP-EP.
8	PFCP-EP sends the Async Sx Modify response message to cnSGW-C service for PDN1. SGW-Service pod waits for the PDN2 Sx Modify response message.
9	UPF2 sends the Sx Modify response (PDN2) message to PFCP-EP.
10	PFCP-EP sends the Async Sx Modify response message to cnSGW-C service for PDN2.
11	The SGW-Service pod sends the following, after receiving the PDN (PDN1, PDN2) responses: • RAB response message to S11-GTP-EP using the gRPC protocol. • Updates to the CDL module
12	SGW-Service pod sends Update Subscriber Context state to CDL, which moves all the bearers to the IDLE state. CDL module updates the information in the database.
13	S11-GTP-EP forwards the RAB response message to MME. MME process the RAB response message.



CHAPTER 8

APN Profile Support

- Feature Summary and Revision History, on page 91
- Feature Description, on page 92
- Feature Configuration, on page 92
- Validation of Uplink Packets for IP Source Violation, on page 94
- Troubleshooting Information, on page 98

Feature Summary and Revision History

Summary Data

Table 26: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Configuration Required
Related Documentation	Not Applicable

Revision History

Table 27: Revision History

Revision Details	Release
Introduced support for IPv6.	2022.04.0
First introduced.	2021.01.0

Feature Description

This feature supports Access Point Name (APN) or Data Network Name (DNN) profile for the SGW (cnSGW-C) service. DNN is equivalent to APN in Evolved Packet System (EPS).

Using the Operator Policy and the Subscriber map, you can determine the DNN Profile for the cnSGW-C service.

Feature Configuration

Configuring this feature involves the following steps:

- Configure DNN Profile. For more information, refer to Configuring DNN Profile, on page 92.
- Configure Network Element Profile. For more information, refer to Configuring Network Element Profile, on page 92.

Configuring DNN Profile

To configure this feature, use the following configuration:

```
config
  profile dnn dnn_name
    upf-selection-policy upf_select_name
      dnn dnn_name network-function-list network_function_list
      end
```

NOTES:

- **dnn** *dnn_name*—Specify the DNN profile name. Must be a string.
- **upf-selection-policy** *upf_select_name*—Specify the UPF selection policy name. Must be a string.
- **network-function-list** *network_function_list*—Specify the list of network functions to which the selected DNN profile is sent. Must be a string.

Configuring Network Element Profile

Network element profile represents peer IP (UPF) profile and has the following configurations:

- Peer address and Port configuration
- Peer-supported DNNs or APNs. This configuration helps in UPF selection.

UPF selection considers priority and capacity parameters.

upf-group-profile indicates the UPF group to which it belongs.

To configure this feature, use the following configuration:

```
config
  profile network-element upf upf_name
    node-id  node_id_value
```

```
n4-peer-address ipv6 ipv6_address
n4-peer-address ipv4 ipv4_address
n4-peer-port port_number
dual-stack-transport { true | false }
dnn-list dnn_list
capacity capacity_value
priority priority_value
upf-group-profile upf_group_name
end
```

NOTES:

- **network-element**—Specify the peer network element.
- **upf** *upf_name*—Specify the UPF peer name.
- **node-id** *node_id_value*—Specify the Node ID of the UPF node.
- **n4-peer-address ipv4** *ipv4_address*—Specify the IPv4 address.
- **n4-peer-address ipv6** *ipv6_address*—Specify the IPv6 address.
- **n4-peer-port** *port_number*—Specify the N4 peer port number. Must be an integer in the range of 0-65535.
- **dual-stack-transport { true | false }**—Enable the dual stack feature that allows you to specify IPv6 or IPv4 address. Specify true to enable this feature.
- **dnn-list** *dnn_list*—Specify the DNN list supported by UPF node.
- **capacity** *capacity_value*—Specify the capacity relative to other UPFs. This is used for load balancing. Must be an integer in the range of 0-65535. Default value is 10.
- **priority** *priority_value*—Specify the static priority relative to other UPFs. This is used for load balancing. Must be an integer in the range of 0-65535. Default value is 1.
- **upf-group-profile** *upf_group_name*—Specify the UPF group profile name. Must be a string.

Configuration Modification Impact

The following table indicates the impact or the configuration change behavior on an existing call, a new PDN, or a new subscriber.

Modification	cnSGW-C Existing Call	cnSGW-C New PDN or New subscriber
Delete the apn-profile	No impact	Applied new configuration based on the changes for the following: • subscriber policy • operator policy • dnn policy • dnn profile

Modification	cnSGW-C Existing Call	cnSGW-C New PDN or New subscriber
Modify the apn profile name in the operator policy	No impact	Applied new configuration based on the changes for the following: • subscriber policy • operator policy • dnn policy • dnn profile

Validation of Uplink Packets for IP Source Violation

Table 28: Feature History

Feature Name	Release Information	Description
Validation of Uplink Packets for IP Source Violation	2024.03.0	When the IP source violation detection feature is enabled, cnSGW validates the invalid packets by checking the source IP address of incoming packets against the valid User Equipment (UE) IP address. The cnSGW derives the UE IP address from the Create Session Request or Response, and includes the IP Source Violation Information Element (IE) to be sent to UPF in the Sx Session Establishment Request Message. This IE indicates the configured action, which is either IGNORE or DISCARD, for the UPF to act on the packets with invalid source IP addresses. This feature enhances the network security and subscriber privacy by preventing the leakage of their data to unauthorized parties. This feature further reduces the risk of legal and regulatory issues for the service provider by complying with the lawful interception requirements. **CLI Introduced**: **ip source-violation [ignore | discard]** in the DNN profile. **Default Setting**: Disabled – Configuration Required to Enable

By default, the IP Source violation detection for uplink packets is disabled. You must enable this feature with either the IGNORE or DISCARD action for the invalid packets. Once enabled, the cnSGW derives the UE

IP address from the Session Create Request or Response and sends it to the UPF. The action that UPF has to take on the packets with invalid source IP addresses is indicated using the IP Source Violation IE in the Sx Session Establishment Request Message.

You can choose between the following options for the IP source violation detection feature depending on the specific requirements and policies of the network operator.

- IGNORE—Choose this option if the network does not have stringent security requirements and ignore the invalid source IP addresses to maintain operational flexibility. Ignoring invalid packets while incrementing statistics allows you to monitor and analyze the occurrence of invalid source IP addresses without immediately discarding the traffic. By forwarding the packets despite invalid source IP addresses, the network can maintain a consistent user experience, in scenarios where the impact of invalid traffic is minimal.
- DISCARD—Choose this option if the network has stringent security requirements and discard packets with invalid source IP addresses to prevent potential security threats, such as IP spoofing. Discarding packets with invalid source IP addresses ensures that only valid traffic is processed and forwarded, maintaining the integrity of the network. By discarding invalid packets, the network avoids potential misrouting that arise from handling packets with incorrect source IP addresses.



Note If this feature is disabled, P Source Violation IE is not included in the Sx Session Establishment Request Message.

Corresponding statistics are incremented for packets with invalid source IP addresses, regardless of whether the packets are ignored or discarded.

You can configure this feature through the **ip source-violation [ignore | discard]** CLI command.

For details on the enabling IP source violation for uplink packets on UPF, see Enablement of IP Source Violation for Uplink Packets in the *UPF Configuration and Admininstration Guide*.

How Validation of Uplink Packets for IP Source Violation Works

The IP Source Violation Information Element (IE) data structure indicates the actions to be taken on packets with invalid source IP addresses from cnSGW to UPF. This IE is part of the Sx Session Establishment Request message and is included when the IP source violation detection feature is enabled.

Table 29: Handling of Uplink Packets with Invalid Source IP Addresses

If Disabled then	If Enabled then

The IP Source Violation IE is not included in the Sx Session Establishment Request message.	Configure one of the following options:
	• IGNORE
	  • Packets with an invalid source IP addresses are validated and then forwarded through the network according to the applicable policy.
	  • Corresponding statistics, which are maintained at cnSGW, are incremented for the detection of the invalid source IP address.
	  • Ignored packets continue to reside in the network traffic flow and are processed as valid packets.
	• DISCARD
	  • Packets with an invalid source IP addresses are identified and then dropped.
	  • Corresponding statistics, which are maintained at cnSGW, are incremented for the action of discarding the packet.
	  • Discarded packets are not processed further by the network device and are not stored, logged, or forwarded.

Enable and Disable Validation of Uplink Packets for IP Source Violation

You can enable or disable IP source violation detection feature in the DNN profile. With the CLI command, you can configure how packets with invalid source IP addresses are handled within the network.

Procedure

Step 1 Log in to the profile DNN mode and enter the DNN profile name.

Example:

```
[sgw] smf(config)# profile dnn <profile name>
```

Step 2 Enter the **ip source-violation** command to configure the IP source violation detection.

Example:

```
[sgw] smf(config)# profile dnn <profile name> ip source-violation
```

Step 3 Choose either the *ignore* or *discard* option for the **ip source-violation** command.

Example:

```
[sgw] smf(config-dnn-intershat)# ip source-violation [ ignore | discard ]
```

To disable IP source violation in the DNN profile, use the **no ip source-violation** CLI command.

If you enter the *ignore* option, the UPF does not check the packets for IP source violation. If you enter the *discard* option, the UPF discards the errant packets.

What to do next

To verify if this feature is configured, use one of the following options:

- Verify Uplink Packet Source Validation on DNN Profile, on page 97
- Verify Uplink Packet Source Validation for NF Service, on page 97

Verify Uplink Packet Source Validation on DNN Profile

Use the procedure in the DNN profile to verify if IP source violation is disabled or enabled with the configured action.

Procedure

Enter the **show running-config profile dnn** *dnn_name* in the DNN profile.

Example:

```
show running-config profile dnn intershat
```

The following is an example output of the **show running-config profile dnn** *dnn_name* CLI command where the *discard* option is configured.

```
show running-config profile dnn intershat
 profile dnn intershat
 dns primary ipv4 209.165.200.229
 dns primary ipv6 66:66:1::aa
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn intershat
 ip source-violation discard
exit
```

Verify Uplink Packet Source Validation for NF Service

Use this procedure to verify the details about the subscriber, such as the IMSI, MEI, MSISDN, and the configuration of the IP source violation feature for a specific subscriber.

Procedure

Enter the **show subscriber nf-service sgw imsi** *imsi_value*.

Example:

```
show subscriber nf-service sgw imsi ABX
```

The following is an example output of the **show subscriber nf-service sgw imsi** *imsi_value* CLI command where the *ignore* option is configured.

```
show subscriber nf-service sgw imsi ABX
subscriber-details
{
  "subResponses": [
    {
      "status": true,
      "genericInfo": {
        "imsi": "imsi-204163553638360",
        "mei": "imeisv-3590730650683317",
        "msisdn": "msisdn-31638577770",
        "accessType": "EUTRAN",
        "plmnId": {
          "mcc": "310",
          "mnc": "260"
        },
        "sgwProfileName": "cn-sgw",
        "unAuthenticatedImsi": "No"
      },
      "s11cInterfaceInfo": {
        "sgwTeid": "[0x1265a5a7] 308651431",
        "sgwIPv4Address": "10.210.81.0",
        "mmeTeid": "[0xa61f5cc] 174192076",
        "mmeIPv4Address": "172.57.38.19"
      },
      "pdnInfoList": {
        "totalPdn": 1,
        "pdnInfo": [
          {
            "pdnId": "PDN-1",
            "apn": "smartsites.t-mobile",
            "attachType": "Initial Attach",
…
…
            "plmnType": "ROAMER",
            "s5ePeerType": "ROAMER",
            "collocatedSub": "NonCollocated"
            "ipSrcViolation": "ignore"
          }
        ]
      }
    }
  ]
}
```

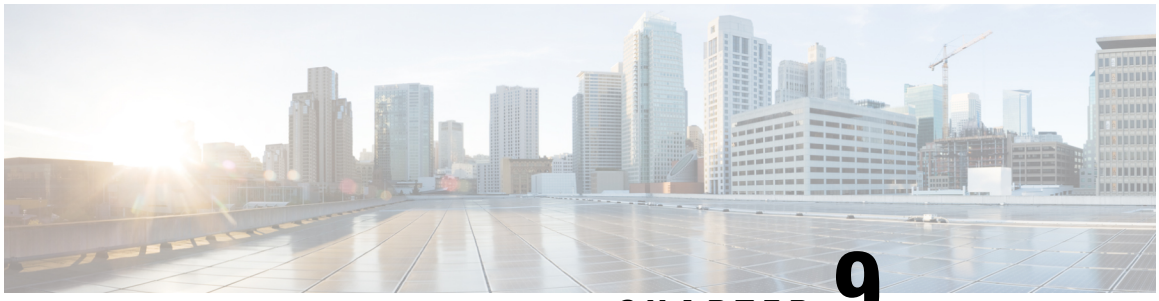
Troubleshooting Information

This section describes troubleshooting information for this feature.

Configuration Errors

This section describes the errors that cnSGW-C might report during the APN profile configuration.

```
show config-error | tab
ERROR COMPONENT   ERROR DESCRIPTION
-------------------------------------------------------------------------------------------------------
SGWProfile        Subscriber policy name : sub_policy in profile sgw1 is not configured
SubscriberPolicy  Operator policy : op_policy1 under subscriber policy sub_policy2 is not
configured
OperatorPolicy    Dnn policy name : dnn_policy1 in operator policy op_policy2 is not
configured
DnnPolicy        Dnn profile name : dnn_profile1 in dnn policy dnn_policy2 is not configured
DnnProfile        UPF selection policy name : upf_sel_policy1 in dnn profile dnn_profile2
is not configured
```



CHAPTER 9

Change Notification Request Handling

- Feature Summary and Revision History, on page 101
- Feature Description, on page 101
- How it Works, on page 102
- OAM Support, on page 104

Feature Summary and Revision History

Summary Data

Table 30: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 31: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

A change notification message is initiated in cnSGW-C to indicate modifications for the User Location Information (ULI) and User CSG Information (UCI) updates. If these updates are valid, the cnSGW-C CDR is initiated. The change notifications may contain the secondary RAT usage IE which is specific to the

cnSGW-C and the ISGW. The cnSGW-C saves the RAT usage information and transmits the usage information in the subsequent CDR message.

Standards Compliance

This feature complies with the following standards specifications:

- *3GPP TS 29.274 "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3"*

How it Works

This section describes how this feature works.

The cnSGW-C network function handles the change notification request using the following approach:

- If the ULI or the UCI changes are valid in the connection request (CNREQ), the associated packet data network (PDN) is updated.
- cnSGW-C initiates a Query URR to get the latest usage information and generates cnSGW-C CDR when:
  - ULI is modified.
  - Charging and ULI trigger is enabled.

For information on configuring charging, see SGW Charging Support chapter.

Call Flows

This section describes the key call flow for this feature.

Change Notification Request Call Flow

This section describes the change notification request and the response call flow.

Figure 20: Change Notification Request Call Flow

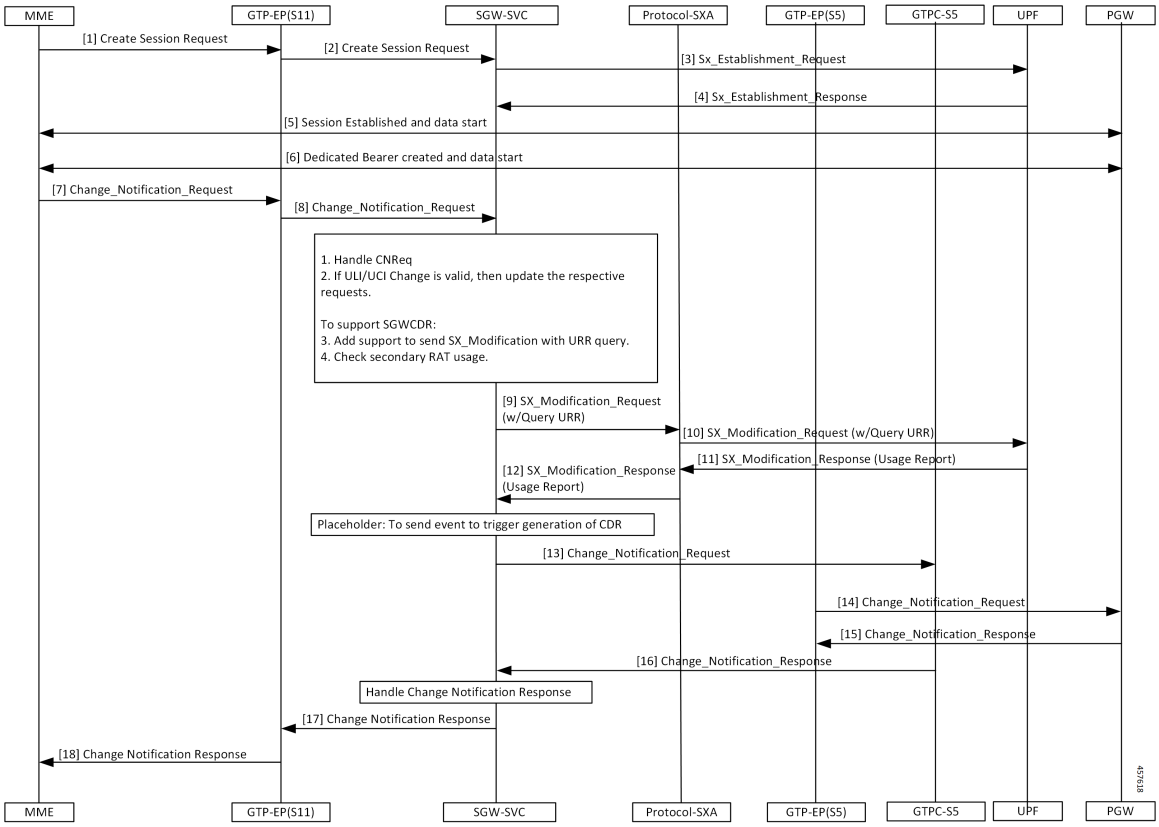


Table 32: Change Notification Request Call Flow Description

Step	Description
1	The MME sends a Create Session Request towards GTP-EP(S11).
2	The GTP-EP(S11) forwards the Create Session Request to the SGW-SVC.
3	The SGW-SVC sends the Sx Establishment Request to the UPF.
4	The UPF responds to the request with the SX Establishment Response directed towards the SGW-SVC.
5	The MME and the PGW establish the sessions and start exchanging data.
6	The MME and the PGW create the dedicated bearer and start exchanging data.
7	The MME sends the Change Notification Request to the GTP-EP.
8	The GTP-EP forwards the Change Notification Request to the SGW-SVC. If ULI or UCI changes are valid in the connection request (CNREQ), the PDN is updated. The GTP-EP sends the Sx Modification Request with the URR query after checking the secondary RAT usage.

Step	Description
9	The SGW-SVC sends the Sx Modification Request with the URR query to the Proto-SXA.
10	The Proto-SXA forwards the Sx Modification Request with the URR query to the UPF.
11	The UPF responds to the request with the Sx Modification Response, with the usage report to the Proto-SXA.
12	The Proto-SXA forwards the Sx Modification Response with the usage report to the SGW-SVC.
13	The SGW-SVC sends the Change Notification Request to the GTPC-S5.
14	The GTPC-S5 forwards the Change Notification Request to the PGW.
15	The PGW responds with the Change Notification Response to the GTPC-S5.
16	The GTPC-S5 forwards the Change Notification Response to the SGW-SVC.
17	The SGW-SVC sends the Change Notification Response to the GTP-EP(S11).
18	The GTP-EP(S11) forwards the Change Notification Response to the MME.

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics Support

The change notification filter displays the status of the change requests for which the notification is invoked. The following are the sample statistics and are provided for reference purposes only.

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="change_notification",status="attempted",sub_fail_reason=""}
1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="change_notification",status="success",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="initial_attach",status="attempted",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="initial_attach",status="success",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="change_notification",status="attempted",sub_fail_reason=""}
1
```

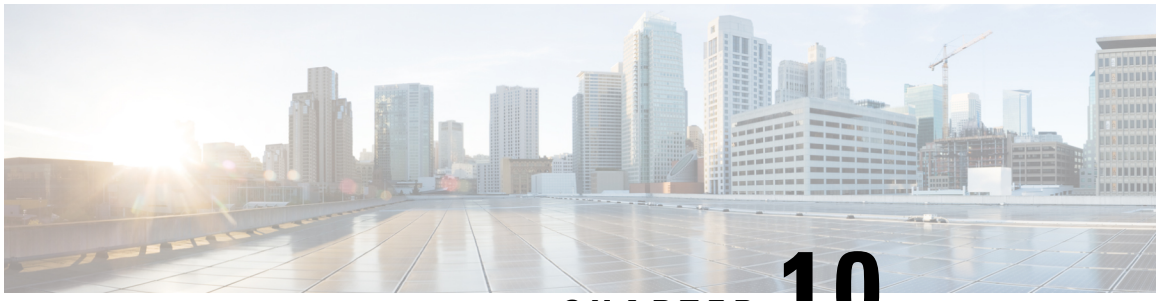
```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="change_notification",status="success",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="initial_attach",status="attempted",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="initial_attach",status="success",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="modify_bearer_req_initial_attach",status="attempted",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service"
,sgw_procedure_type="modify_bearer_req_initial_attach",status="success",sub_fail_reason=""}
 1
```



CHAPTER 10

Clear Subscriber Request

- Feature Summary and Revision History, on page 107
- Feature Description, on page 107
- How it Works, on page 108

Feature Summary and Revision History

Summary Data

Table 33: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 34: Revision History

Revision Details	Release
First introduced.	2020.04

Feature Description

cnSGW-C handles the Clear Subscriber or the PDN Request from the Ops Center.

The Clear Subscriber Request initiates the administrative clearing of subscribers for a specific IMSI or all IMSIs using the local purge and remote signaling procedures.

Based on the OAM query, the cnSGW-C receives the Subscriber Notification message at REST-EP and triggers the Clear Subscriber Request message towards the SGW-Service.

Standards Compliance

This feature complies with the following standards specifications:

- *3GPP TS 23.401 "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access"*
- *3GPP TS 23.214 "Architecture enhancements for control and user plane separation of EPC nodes"*
- *3GPP TS 29.274 "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3"*
- *3GPP TS 29.244 "Interface between the Control Plane and the User Plane nodes"*

How it Works

This section describes how this feature works.

When the cnSGW-C receives the admin-initiated Deletion Request with the purge option as “true”, it initiates Sx signaling towards User Plane and exchanges following messages:

1. SGW sends a Sx Session Deletion Request to User Plane.
2. User Plane sends a Sx Session Deletion Response SGW.

When cnSGW-C receives the Deletion Request with the purge option as “false”, it performs the Sx signaling towards User Plane and GTP-C signaling towards MME and PGW. The cnSGW-C exchanges the following messages with User Plane, MME, and PGW:

1. SGW sends the Sx Session Modification Request to the User Plane.
2. User Plane sends the Sx Session Modification Response to SGW.
3. SGW sends the Delete Bearer Request to MME.
4. SGW sends the Delete Session Request to PGW.
5. MME sends the Delete Bearer Response to SGW.
6. PGW sends the Delete Session Response to SGW.
7. SGW sends the Sx Session Deletion Request to User Plane.
8. User Plane sends the Sx Session Deletion Response to SGW.

cnSGW-C sends the Delete Session Request towards PGW and Delete Bearer Request towards MME. After receiving the response from both remote peers, the cnSGW-C sends Sx Session Deletion Request towards User Plane to clear the sessions.

Supported Clear Command

cnSGW-C supports the following clear commands:

Table 35: Supported Clear Commands

Supported Clear Command Options	GTP-C Signalling (Towards MME/PGW)	Sx Signalling (Towards UP)	Impact (Subscriber/PDN)
clear sub all **clear sub all purge false**	Yes	Yes	All subscribers
clear sub all purge true	No	Yes	All subscribers
• **clear sub namespace sgw imsi** *imsi_val* • **clear sub namespace sgw imsi***imsi_val***purge false**	Yes	Yes	Subscriber with IMSI as *imsi_val*
clear sub namespace sgw imsi*imsi_val***purge true**	No	Yes	Subscriber with IMSI as *imsi_val*
clear sub namespace sgw imsi*imsi_val***ebi***ebi_value*	Yes	Yes	PDN with IMSI as *imsi_val* and default ebi as *ebi_value*

Call Flows

This section describes the key call flows for this feature.

Clear PDN Call Flow

This section describes the Clear PDN call flow.

Figure 21: Clear PDN Call Flow

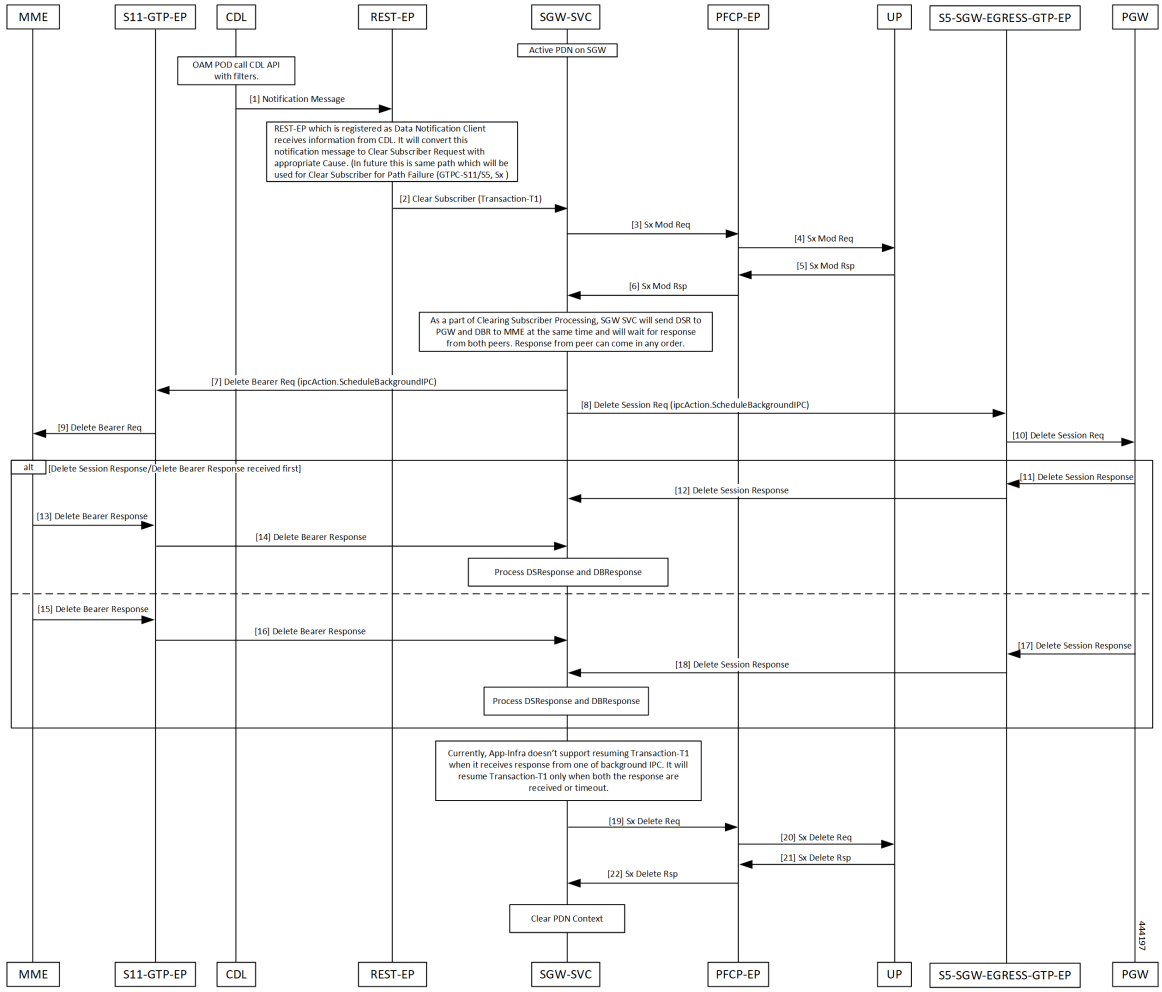
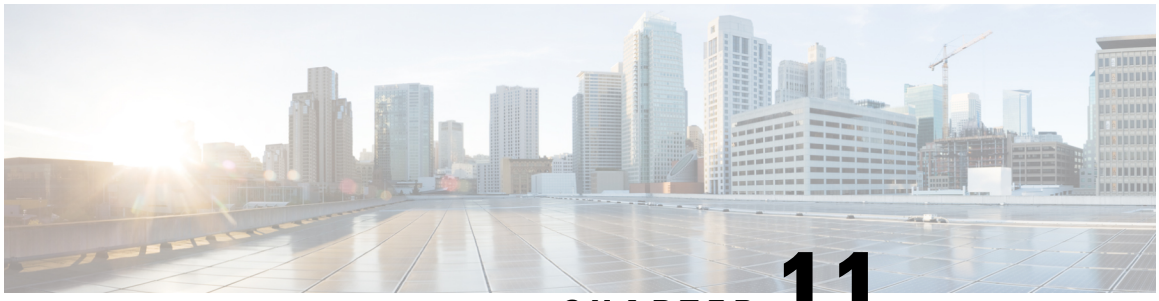


Table 36: Clear PDN Call Flow Description

Step	Description
1	The OAM pod calls the CDL API with the filters. CDL sends the notification message to REST EP.
2	The REST-EP converts this message to Clear Subscriber Request with a cause and sends Clear Subscriber to the SGW-Service pod. Transaction-T1 started.
3-6	The SGW-Service pod sends Sx Modification Request to UPF through PFCP-EP. The SGW-Service pod receives Sx Modification Response from UPF through PFCP-EP.
7	The SGW-Service pod sends the Delete Bearer Request to the S11-GTP-EP.
8	The SGW-Service pod sends the Delete Session Request to the S5-SGW-EGRESS-GTP-EP.

Step	Description
9	The S11-GTP-EP sends the Delete Bearer Request to MME.
10-12	The S5-SGW-EGRESS-GTP-EP sends the Delete Session Request to PGW. The PGW sends the Delete Session Response to S5-SGW-EGRESS-GTP-EP. The S5-SGW-EGRESS-GTP-EP forwards this request to the SGW-Service pod.
13-16	MME sends the Delete Bearer Response to S11-GTP-EP. S11-GTP-EP forwards to the SGW-Service pod.
17, 18	PGW sends the Delete Session Response to S5-SGW-EGRESS-GTP-EP. S5-SGW-EGRESS-GTP-EP forwards this request to the SGW-Service pod.
19-22	The SGW-Service pod sends the Sx Delete Request to PFCP-EP. The PFCP-EP forwards the request to UPF. UPF sends the Sx Delete Response to PFCP-EP, which it forwards it to the SGW-Service pod.



CHAPTER 11

Context Replacement Support

- Feature Summary and Revision History, on page 113
- Feature Description, on page 114
- How it Works, on page 114
- OAM Support, on page 119

Feature Summary and Revision History

Summary Data

Table 37: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 38: Revision History

Revision Details	Release
Introduced support for partial context replacement.	2021.02.0
First introduced.	2020.01.0

Feature Description

The cnSGW-C supports context replacement when it receives Create Session Request (CSReq) with the existing EBI. When the MME node and cnSGW-C are not synchronized, the session gets locally terminated on the MME. The MME sends a CSReq with the EBI that is already present in the cnSGW-C. If the CSReq contains a TEID with value as non-ZERO, then cnSGW-C partially replaces the context. When TEID is zero, cnSGW-C performs full context replacement.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Full Context Replacement Call Flow

This section describes the full context replacement call flow.

Create Session Request Call Flow

This section describes the Create Session Request call flow.

Figure 22: Create Session Request (Context Replacement – Single or Multi-PDN subscriber) Call Flow

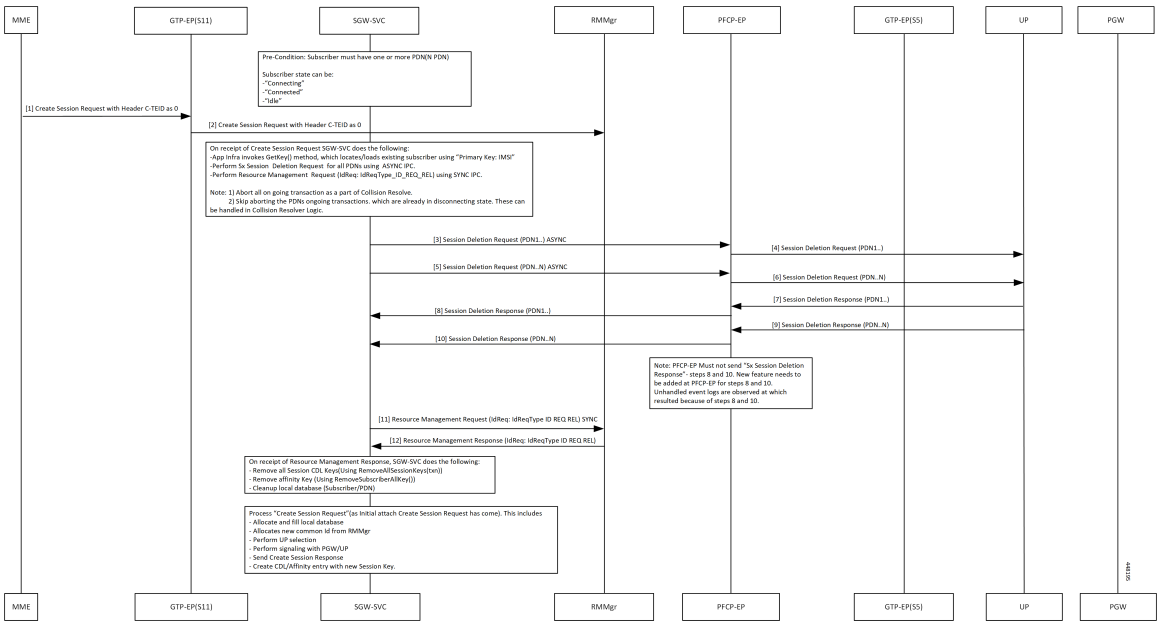


Table 39: Create Session Request (Context Replacement – Single or Multi-PDN subscriber) Call Flow Description

Step	Description
1	MME sends Create Session Request with C-TEID as zero to GTPC-EP ingress.
2	GTPC-EP ingress forwards the Create Session Request to SGW-SVC. Following actions takes place: • App Infra invokes the GetKey() method, which locates and loads the existing subscribers using Primary Key: IMSI. • Performs Sx Session Deletion Request for all PDNs using ASYNC IPC • Performs Resource Management Request (IdReq: IdReqType_ID_REQ_REL) using SYNC IPC
3, 5	The SGW service pod sends the Delete Session Request for PDN 1 - N to PFCP-EP.
4, 6	PFCP-EP forwards Delete Session Request for PDN 1 - N to UPF.
7, 9	PFCP-EP receives Delete Session Response for PDN 1 to N from UPF.
8, 10	PFCP-EP forwards Delete Session Response for PDN 1 - N to SGW service pod.
11	SGW service pod sends Resource Management Request to RMMgr with request ID-type as Request REL.
12	SGW service pod receives Resource Management Response from RMMgr with Req ID-type as REQ REL. The SGW service pod performs following: • Removes all session CDL keys (Using RemoveAllSessionKeys(txn)) • Removes affinity Key (Using RemoveSubscriberAllKey()) • Cleans up the local database (Subscriber/PDN)

Note You can ignore unhandled events for the Deletion Response from UPF.

Partial Context Replacement Call Flow

This section describes the partial context replacement call flow.

When cnSGW-C receives a CSReq with the existing EBI and TEID as non-ZERO, then cnSGW-C performs a partial context replacement by invoking the following call flows:

- EBI received in CSReq is for the existing default bearer.
- EBI received in CSReq is for the existing dedicated bearer.

Create Session Request with Default Bearer EBI Call Flow

This section describes the Create Session Request with Default Bearer EBI call flow.

Figure 23: CSReq with Default Bearer EBI Call Flow

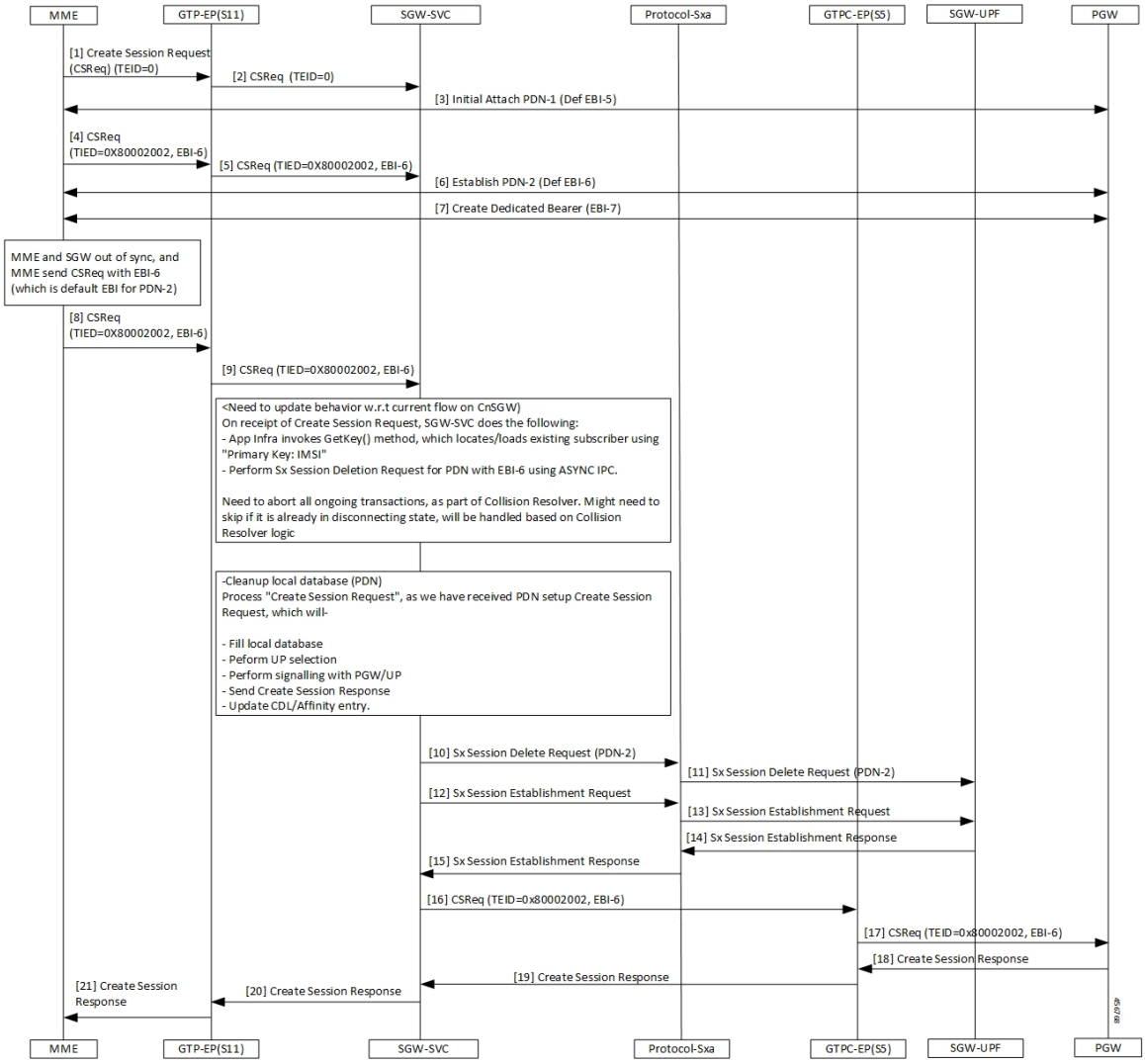


Table 40: CSReq with Default Bearer EBI Call Flow Description

Step	Description
1	The MME sends a Create Session Request with TIED value as 0 to the GTPC-EP(S11).
2	The GTPC-EP(S11) forwards the Create Session Request with TIED value as 0 to the SGW-SVC.
3	The MME and the PGW process the Initial Attach PDN-1 with the default EBI-5 process.
4	The MME sends a Create Session Request TIED=0x80002002 with EBI-6 to the GTPC-EP.

Step	Description
5	The GTPC-EP forwards the Create Session Request TIED=0x80002002 with EBI-6 to the SGW-SVC.
6	The MME and the PGW establish the PDN-2 with default EBI-6 connection.
7	The MME and PGW complete the Create Dedicated Bearer with EBI-7 process.
8	If the SGW and MME are not in sync, the MME sends a Create Session Request with EBI-6 present in the SGW.
9	The GTPC-EP sends a CSReq TIED= 0x80002002 with EBI-6 to SGW.
10	After receiving the Create Session Request, the SGW-SVC performs the following- • Cleans up the PDN with default EBI=6. • Sends the Sx signalling to UPF to clear the session. • Performs the Create Session Request as a new PDN-Setup. The SGW sends an Sx Session Delete Request on PDN-2 to Protocol-SXA.
11	The Protocol-SXA forwards a Sx Session Delete Request to SGW-UPF.
12	The SGW sends a Session Establishment Request to the Protocol-SXA.
13	The Protocol-SXA forwards a Sx Session Establishment Request to SGW-UPF.
14	The SGW-UPF responds to the Protocol-SXA with the Sx Session Establishment Response.
15	The Protocol-SXA sends the Sx Session Establishment Response to the SGW-SVC.
16	The SGW-SVC sends the Create Session Request TIED= 0x80002002 with EBI-6 to the GTPC-EP.
17	The GTPC-EP sends the Create Session Request TIED= 0x80002002 with EBI-6 to the PGW.
18	The PGW sends a Create Session Response to the GTPC-EP.
19	The GTPC-EP responds to the SGW-SVC with the Create Session Response.
20	The SGW-SVC forwards the response to the GTPC-EP.
21	The GTPC-EP sends the Create Session Response to the MME.

Create Session Request with Dedicated Bearer EBI Call Flow

This section describes the Create Session Request with the Dedicated EBI call flow.

Figure 24: CSReq with Dedicated Bearer EBI Call Flow

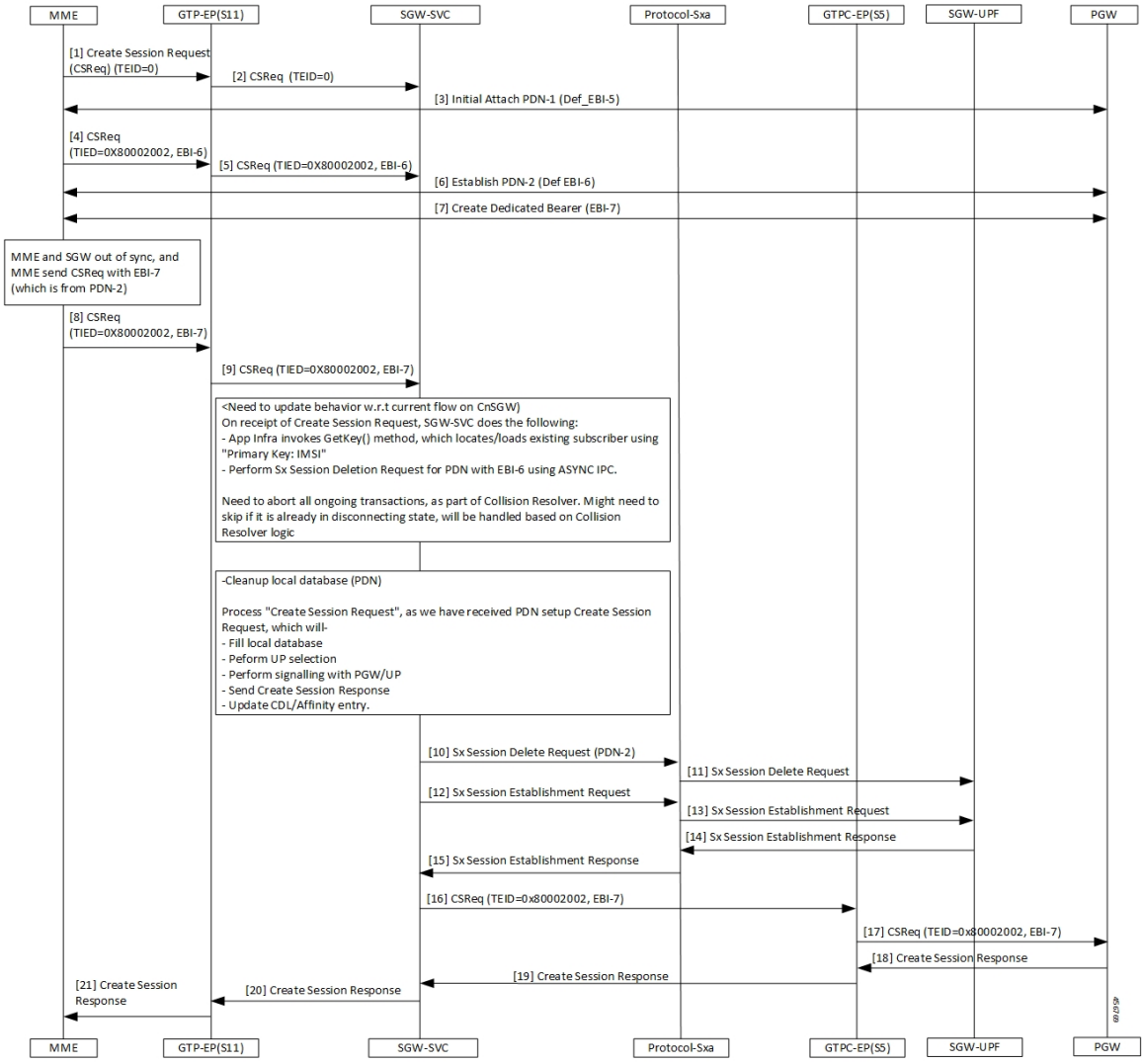


Table 41: CSReq with Dedicated Bearer EBI Call Flow Description

Step	Description
1	The MME sends a Create Session Request with the TIED value as zero to the GTPC-EP(S11).
2	The GTPC-EP(S11) forwards the Create Session Request with TIED value as zero to the SGW-SVC.
3	The MME and the PGW process the Initial Attach PDN with the EBI-5 process.
4	The MME sends the Create Session Request with EBI-6 to the GTPC-EP.
5	The GTPC-EP forwards the Create Session Request with EBI-6 to the SGW-SVC.
6	The MME and PGW establish the PDN with the EBI-6 connection.

Step	Description
7	The MME and PGW complete the Create Dedicated Bearer with EBI-7 process.
8	If the SGW and MME are not in sync, then MME sends a Create Session Request with EBI-6 present in SGW.
9	The GTPC-EP sends a CSReq with EBI-7 to SGW.
10	After receiving the Create Session Request, the SGW-SVC: • Cleans up the PDN with default EBI=6. • Sends the Sx signalling to UPF to clear the session. • Performs the Create Session Request as new PDN-Setup. The SGW sends a Sx Session Delete Request on PDN-2 to Protocol-SXA.
11	The Protocol-SXA forwards the Sx Session Delete Request to SGW-UPF.
12	The SGW sends a Session Establishment Request to the Protocol-SXA.
13	The Protocol-SXA forwards a Sx Session Establishment Request to SGW-UPF.
14	The SGW-UPF responds to the Protocol-SXA with the Sx Session Establishment Response.
15	The Protocol-SXA sends the Sx Session Establishment Response to the SGW-SVC.
16	The SGW-SVC sends the Create Session Request containing TIED=0x80002002, EBI-7 to the GTPC-EP.
17	The GTPC-EP sends the Create Session Request containing TIED=0x80002002, EBI-7 to the PGW.
18	The PGW sends a Create Session Response to the GTPC-EP.
19	The GTPC-EP responds to the SGW-SVC with the Create Session Response.
20	The SGW-SVC forwards the response to the GTPC-EP.
21	The GTPC-EP sends the Create Session Response to the MME.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics

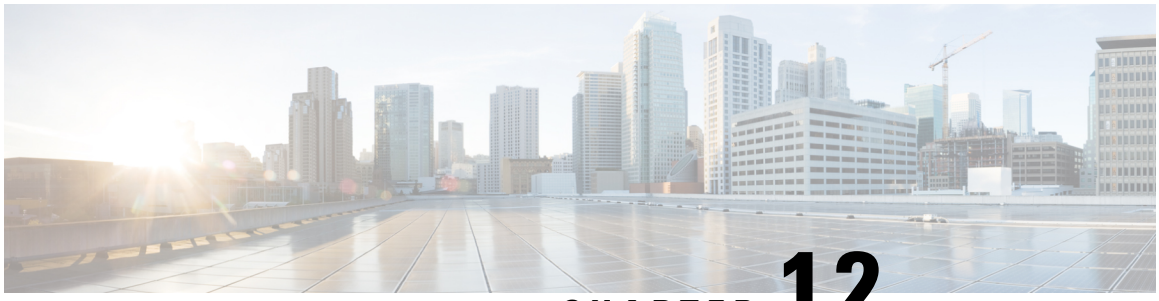
The following statistics are supported for the partial context replacement feature.

- sgw_pdn_disconnect_stats: Captures the total number of SGW PDN in the disconnected status.

  An example of the Prometheus query:

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="cn",data_center=\

"cn",instance_id="0",pdn_type="ipv4",rat_type="EUTRAN",reason="context_replacement",\
service_name="sgw-service"} 1
```



CHAPTER 12

Dedicated Bearer Support

- Feature Summary and Revision History, on page 121
- Feature Description, on page 121
- Setup and Update Dedicated Bearers, on page 122
- Delete Dedicated Bearers, on page 129

Feature Summary and Revision History

Summary Data

Table 42: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 43: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

Setup and Update Dedicated Bearers

cnSGW-C supports creating and updating single/multiple dedicated bearers.

Delete Dedicated Bearers

cnSGW-C supports deletion of single/multiple dedicated bearers.

Setup and Update Dedicated Bearers

Feature Description

cnSGW-C supports creating and updating dedicated bearers for both single and multiple PDN subscribers. It also supports multiple bearer contexts as part of single create bearer procedure.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Dedicated Bearer Setup – Request Accepted Call Flow

This section describes the Dedicated Bearer Setup – Request Accepted call flow.

Figure 25: Dedicated Bearer Setup – Request Accepted Call Flow

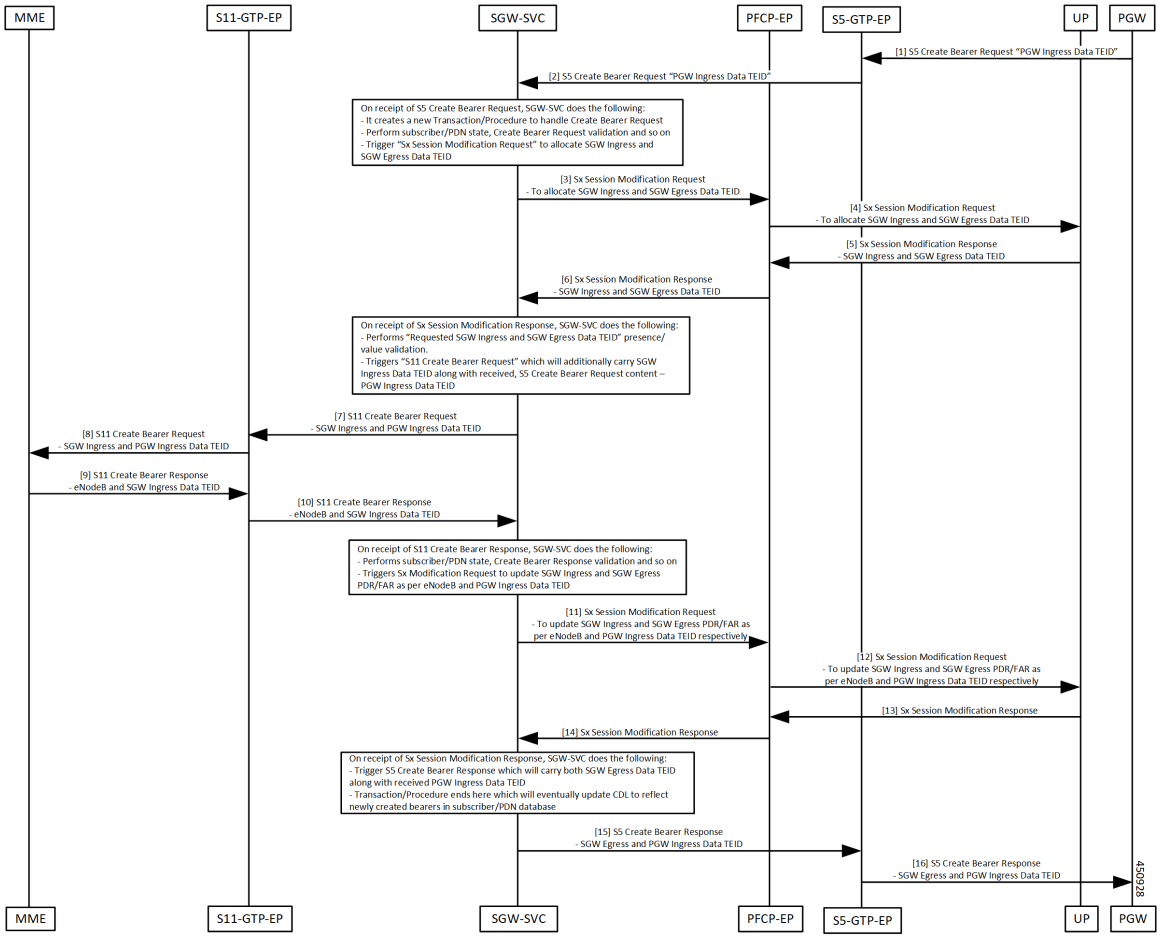


Table 44: Dedicated Bearer Setup – Request Accepted Call Flow Description

Step	Description
1	The PGW sends the S5 Create Bearer Request to the S5-GTPC-EP pod.
2	The S5-GTPC-EP pod forwards the S5 Create Bearer Request to the SGW-SVC pod.
3	The SGW-SVC receives the S5 Create Bearer request and performs the following: • Creates a new transaction • Performs GTP validations • Triggers the Sx Modification Request to the PFCP-EP pod
4	The PFCP-EP pod forwards the Sx Modification Request to the UP for allocating SGW Ingress and SGW Egress TEIDs.
5	The PFCP-EP pod receives the Sx Modification Response with SGW Ingress and SGW Egress TEIDs, from the UP.

Step	Description
6	The PFCP-EP pod forwards the Sx Modification Response with SGW Ingress and SGW Egress TEIDs, to the SGW-SVC.
7	The SGW-SVC receives the Sx Modification response and performs the following: • Validates the received SGW Ingress and SGW Egress TEIDs • Triggers the S11 Create Bearer Request with the SGW Ingress TEID to the S11-GTPC-EP pod
8	The S11-GTPC-EP pod forwards the S11 Create Bearer Request with the SGW Ingress TEID, to the MME.
9	The MME sends the S11 Create Bearer Response to the S11-GTPC-EP pod.
10	The S11-GTPC-EP pod forwards the S11 Create Bearer Response to the SGW-SVC.
11	The SGW-SVC receives the S11 Create Bearer response and performs the following: • GTP validations • Triggers the Sx Modification Request to the PFCP-EP pod to update SGW Ingress and SGW Egress PDR/FAR, with the MME and the PGW GTPU-TEID
12	The PFCP-EP pod forwards the Sx Modification Request to the UP.
13	The UP sends the Sx Modification Response to the PFCP-EP pod.
14	The PFCP-EP pod forwards the Sx Modification Response to the SGW-SVC pod.
15, 16	The SGW-SVC pod receives the Sx Modification Response and performs the following: • Ends the transaction/procedure • Updates the CDL • Sends the S5 Create Bearer Response with SGW Egress TEIDs with matching PGW GTPU TEIDs, and with the cause as Accepted.

Dedicated Bearer Setup – Request Accepted Partially Call Flow

This section describes the Dedicated Bearer set up call flow. In this procedure, the MME sends the Create Bearer Response with the GTP cause as Request Accepted Partially.

Prerequisite: Create Bearer Procedure with two bearer contexts.

Figure 26: Dedicated Bearer Setup – Request Accepted Partially Call Flow

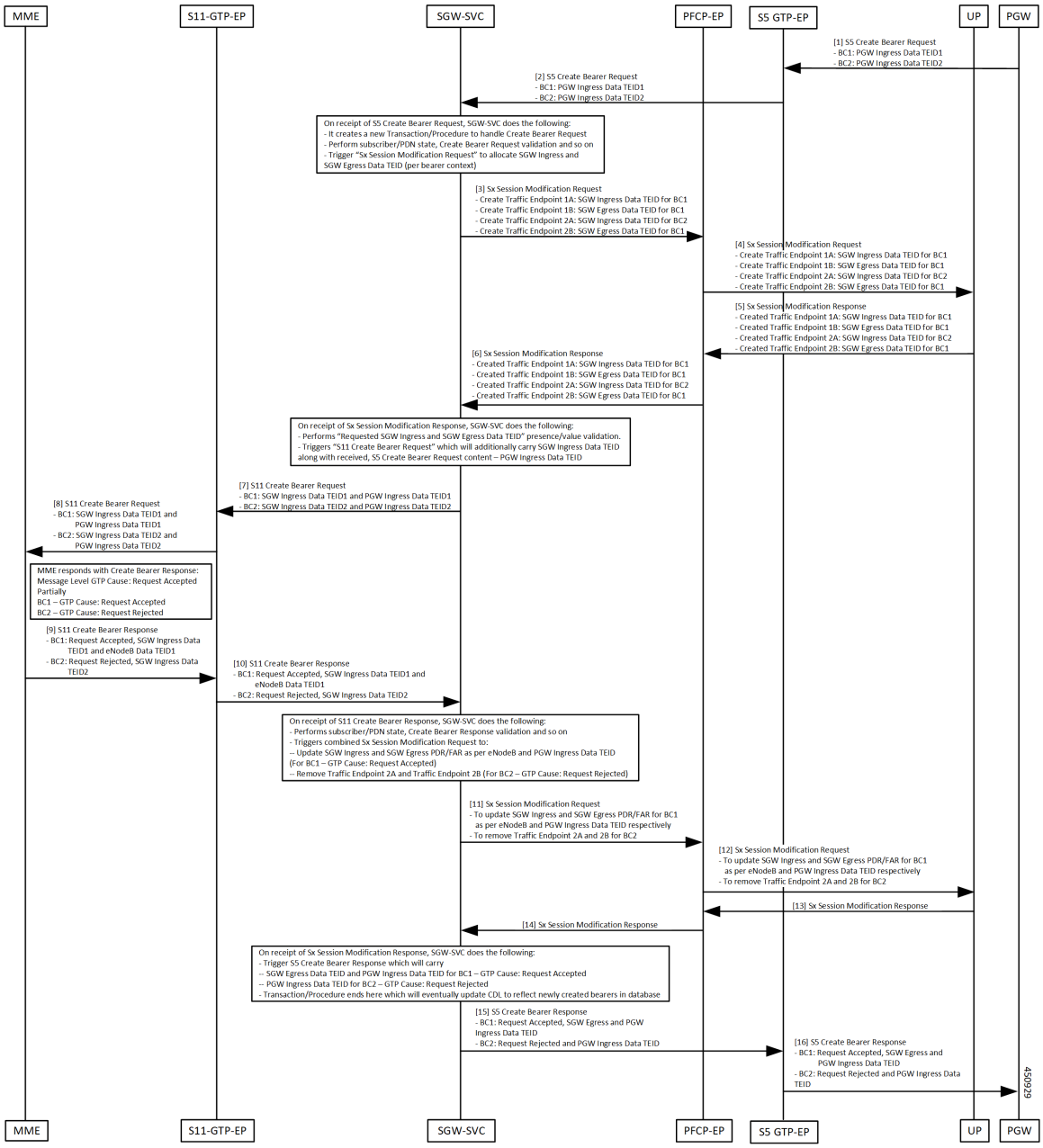


Table 45: Dedicated Bearer Setup – Request Accepted Partially Call Flow Description

Step	Description
1	The PGW sends the S5 Create Bearer Request with multiple bearer contexts to the S5-GTPC-EP pod.
2	The S5-GTPC-EP pod forwards the S5 Create Bearer Request to the SGW-SVC pod.

Step	Description
3	The SGW-SVC pod receives the S5 Create Bearer request and performs the following: • Creates a new transaction • Performs GTP validations • Triggers the Sx Modification Request to allocate SGW Ingress and SGW Egress TEIDs to the PFCP-EP pod.
4	The PFCP-EP pod forwards the Sx Modification Request to the UP.
5	The UP sends the Sx Modification Response to the PFCP-EP pod.
6	The PFCP-EP pod forwards the Sx Modification Response to the SGW-SVC pod.
7	The SGW-SVC receives the Sx Modification Response and performs the following: • Validates the received SGW Ingress and SGW Egress TEIDs • Triggers an S11 Create Bearer Request with the SGW Ingress TEID to the S11-GTPC-EP pod
8	The S11-GTPC-EP pod forwards the S11 Create Bearer Request to the MME.
9	The S11-GTPC-EP receives the S11 Create Bearer Response from the MME, with the Message Level GTP cause as Request Accepted Partially: • For some Bearer Contexts, GTP cause is Request Accepted • For some Bearer Contexts, GTP cause is Request Rejected
10	The S11-GTPC-EP pod forwards the S11 Create Bearer Response to the SGW-SVC pod.
11	The SGW-SVC pod receives the S11 Create Bearer response and performs the following: • GTP validations • For successful bearers: Triggers the Sx Modification Request to the PFCP-EP pod for updating SGW Ingress and SGW Egress PDR/FAR with the MME and the PGW GTPU-TEID • For failed bearers: Removes the traffic endpoints
12	The PFCP-EP pod forwards the Sx Modification Request to the UP.
13	The UP sends the Sx Modification Response to the PFCP-EP pod.
14	The PFCP-EP pod forwards the Sx Modification Response to the SGW-SVC pod.

Step	Description
15, 16	The SGW-SVC pod receives the Sx Modification Response and performs the following: • Ends the transaction/procedure • Updates the CDL • For successful bearers: Sends the S5 Create Bearer Response with SGW Egress TEIDs with matching PGW GTPU TEIDs. • For failed bearers: Sends the bearer contexts as is with the message level cause as Partially Accepted.

Dedicated Bearer Update – Request Accepted Call Flow

This section describes the Default/Dedicated Bearer Update Procedure call flow.

Single Update Bearer Procedure supports:

- Default bearer QoS/TFT change
- Single/Multiple dedicated bearer QoS/TFT change
- APN-AMBR change

Note The call flow doesn't contain Sx Communication Messages related to the Default/Dedicated Bearer Update procedure.

Figure 27: Default/Dedicated Bearer Update (Single/Multiple Bearers) Support Call Flow

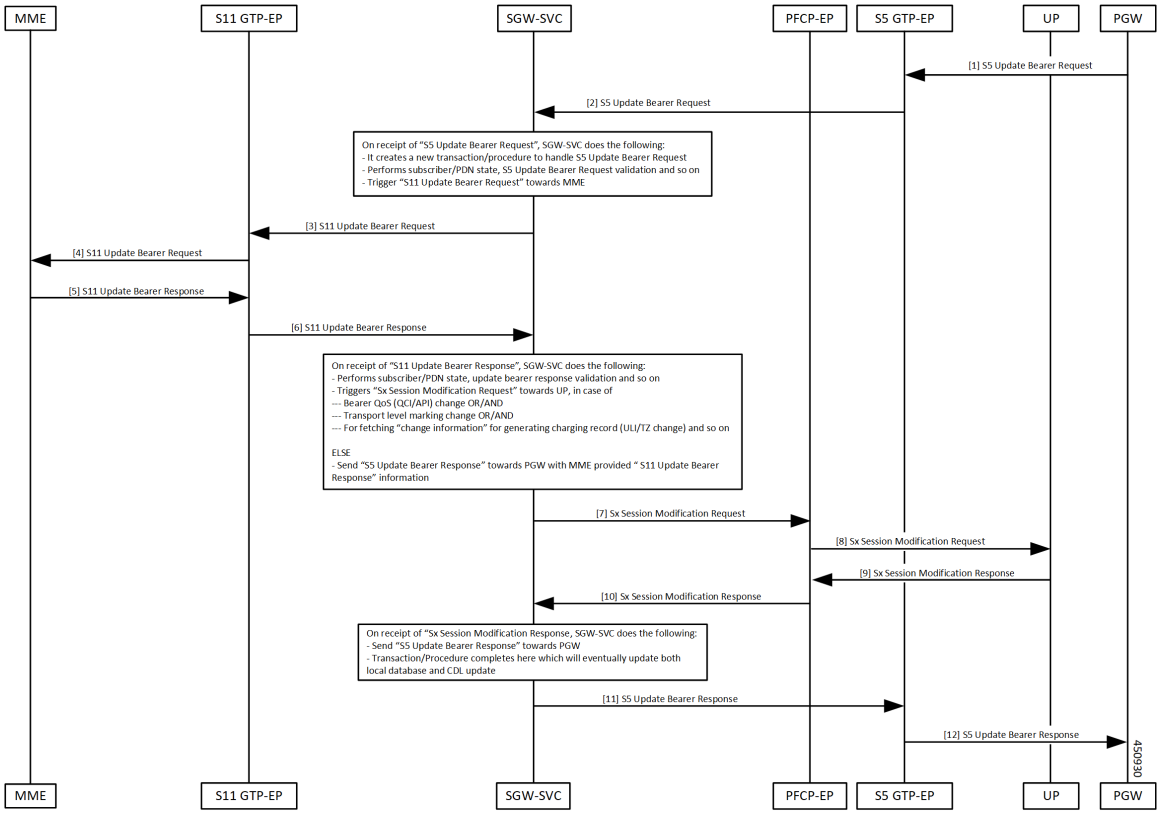


Table 46: Default/Dedicated Bearer Update (Single/Multiple Bearers) Support Call Flow Description

Step	Description
1	The PGW sends the S5 Update Bearer Request with multiple bearer contexts to the GTPC-EP pod.
2	The GTPC-EP pod forwards the S5 Update Bearer request to the SGW-SVC pod.
3	SGW-SVC receives the S5 Update Bearer request and performs the following: • Creates a new transaction • Performs GTP validations • Triggers the S11 Update Bearer Request to the GTPC-EP pod
4	The GTPC-EP pod forwards the S11 Update Bearer Request to the MME.
5	The MME sends the S11 Update Bearer Response to the GTPC-EP pod.
6	The GTPC-EP pod forwards the S11 Update Bearer Response to the SGW-SVC pod.

Step	Description
7	SGW-SVC receives the S11 Update Bearer Response and performs GTP validations. • If: Any of the following is true, the SGW-SVC triggers Sx Modification Request to the PFCP-EP pod: • Bearer QoS (QCI/ARP) change • Transport Level Marking change • Fetch charging information for generating charging record ULI/TZ change • Else: The SGW-SVC sends the S11 Update Bearer Response to the PGW with the MME-provided S11 Update Bearer Response information.
8	The PFCP-EP pod forwards the Sx Session Modification Request to the UP.
9	The UP sends the Sx Session Modification Response to the PFCP-EP pod.
10	The PFCP-EP pod forwards the Sx Session Modification Response to the SGW-SVC. The SGW-SVC receives the Sx Modification Response and performs the following: • Ends the transaction/procedure • Updates the CDL • Sends the S5 Update Bearer Response with cause as Accepted
11	The SGW-SVC sends the S5 Update Bearer Response to the PFCP-EP pod. The PFCP-EP pod forwards the S5 Update Bearer Response to the GTP-EP pod.
12	The GTP-EP pod forwards the S5 Update Bearer Response to the UP. The UP forwards the S5 Update Bearer Response to the PGW.

Delete Dedicated Bearers

Feature Description

cnSGW-C supports single/multiple dedicated bearer deletion as part of single delete bearer procedure.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Dedicated Bearer Deletion Procedure Call Flow

This section describes the Dedicated Bearer Delete Procedure call flow.

Figure 28: Dedicated Bearer Deletion Procedure (Single/Multiple Bearer) Call Flow

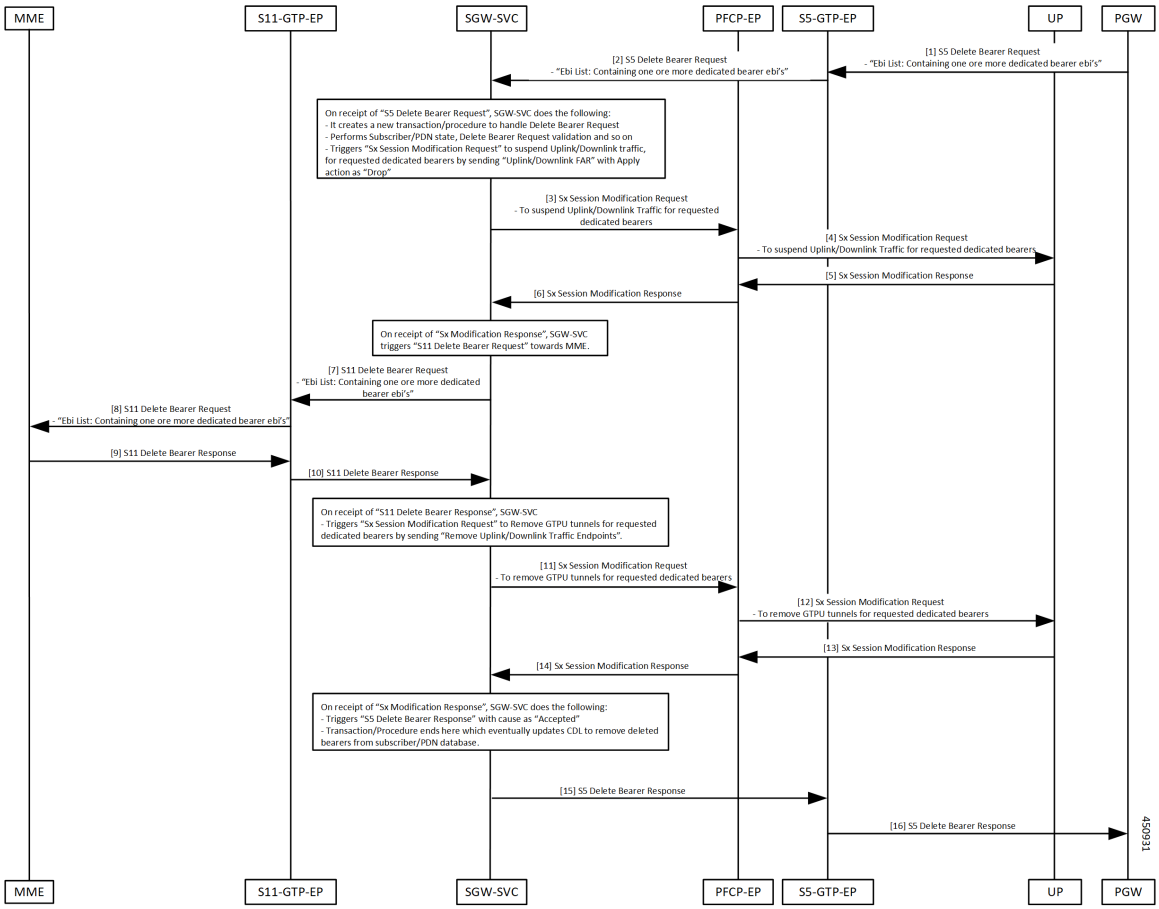
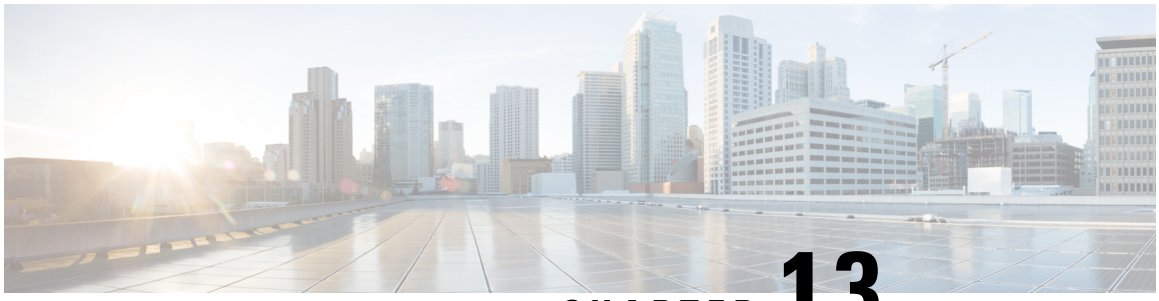


Table 47: Dedicated Bearer Deletion Procedure (Single/Multiple Bearer) Call Flow Description

Step	Description
1	The PGW sends the S5 Delete Bearer Request with EBI list containing one or more dedicated bearer EBIs, to the GTPC-EP pod.
2	The GTPC-EP pod forwards the S5 Delete Bearer Request to the SGW-SVC pod.
3	The SGW-SVC pod receives the S5 Delete Bearer request and performs the following: • Creates a new transaction • Performs GTP validations • Triggers the Sx Modification Request to the PFCP-EP pod to suspend uplink/downlink traffic for the requested bearers
4	The PFCP-EP pod forwards the Sx Modification Request to the UP.

Step	Description
5	The UP sends the Sx Modification Response to the PFCP-EP pod.
6	The PFCP-EP pod forwards the Sx Modification Response to the SGW-SVC pod.
7	The SGW-SVC pod receives the Sx Modification Response and triggers the S11 Delete Bearer Request to the GTPC-EP pod.
8	The GTPC-EP pod forwards the S11 Delete Bearer Request to the MME.
9	The MME sends the S11 Delete Bearer Response to the GTPC-EP pod.
10	The GTPC-EP pod forwards the S11 Delete Bearer Response to the SGW-SVC pod.
11	The SGW-SVC receives the S11 Delete Bearer Response and triggers the Sx Modification Request to the PFCP-EP pod, to remove traffic endpoints for removal of the GTPU tunnels for the requested dedicated bearers.
12	The PFCP-EP pod forwards the Sx Modification Request to the UP to remove GTP tunnels for the requested dedicated bearers.
13	The UP sends the Sx Modification Response to the PFCP-EP pod.
14	The PFCP-EP forwards the Sx Modification Response to the SGW-SVC pod.
15	The SGW-SVC receives the Sx Modification Response and performs the following: • Ends the transaction/procedure • Updates the CDL • Sends the S5 Delete Bearer Response with cause as Accepted, to the GTP-EP pod
16	The GTP-EP pod forwards the S5 Delete Bearer Response to the PGW.



CHAPTER 13

Delete Bearer and Delete Session Request

- Feature Summary and Revision History, on page 133
- Feature Description, on page 133
- How it Works, on page 134

Feature Summary and Revision History

Summary Data

Table 48: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 49: Revision History

Revision Details	Release
First introduced.	2020.04

Feature Description

This feature supports the following:

- Deletion of Session Request from the MME
- Deletion of Bearer Request from the PGW

This deletion helps in clearing the PDN connection at the SGW, which in turn clears resources at the cnSGW-C, and releases all the relevant TEIDs.

Delete from MME

1. cnSGW-C sends the Sx Modification Request to the User Plane (UP) to mark the forwarding action as DROP so that all uplink or downlink packets are dropped at the SGW-U.
2. cnSGW-C sends the Delete Session Request to the PGW/SMF.
3. After SGW receives the Delete Session Response from the PGW/SMF, cnSGW-C sends the Sx Terminate Request to the UP to clear the session.
4. After UP confirms the deletion of the SGW-U session, cnSGW-C releases the allocated ID by sending request to the Node Manager, and the Delete Session Response to the MME.

Delete from PGW

1. cnSGW-C sends the Sx Modification Request to the UP to mark the forwarding action as DROP so that all the uplink and downlink packets are dropped at the SGW-U.
2. cnSGW-C sends the Delete Bearer Request to the MME.
3. After SGW receives the Delete Bearer Response from the MME, the cnSGW-C sends the Sx Terminate Request to the UP to clear the session.
4. After UP confirms the deletion of the SGW-U session, cnSGW-C releases the allocated ID by sending request to the Node Manager, and the Delete Bearer Response to the PGW.

Standard Compliance

The Delete Bearer and Delete Session Request Support feature complies with the following standards:

- *3GPP TS 23.401 "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access"*
- *3GPP TS 23.214 "Architecture enhancements for control and user plane separation of EPC nodes"*
- *3GPP TS 29.274 "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3"*
- *3GPP TS 29.244 "Interface between the Control Plane and the User Plane nodes"*

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Figure 29: Delete from MME Call Flow

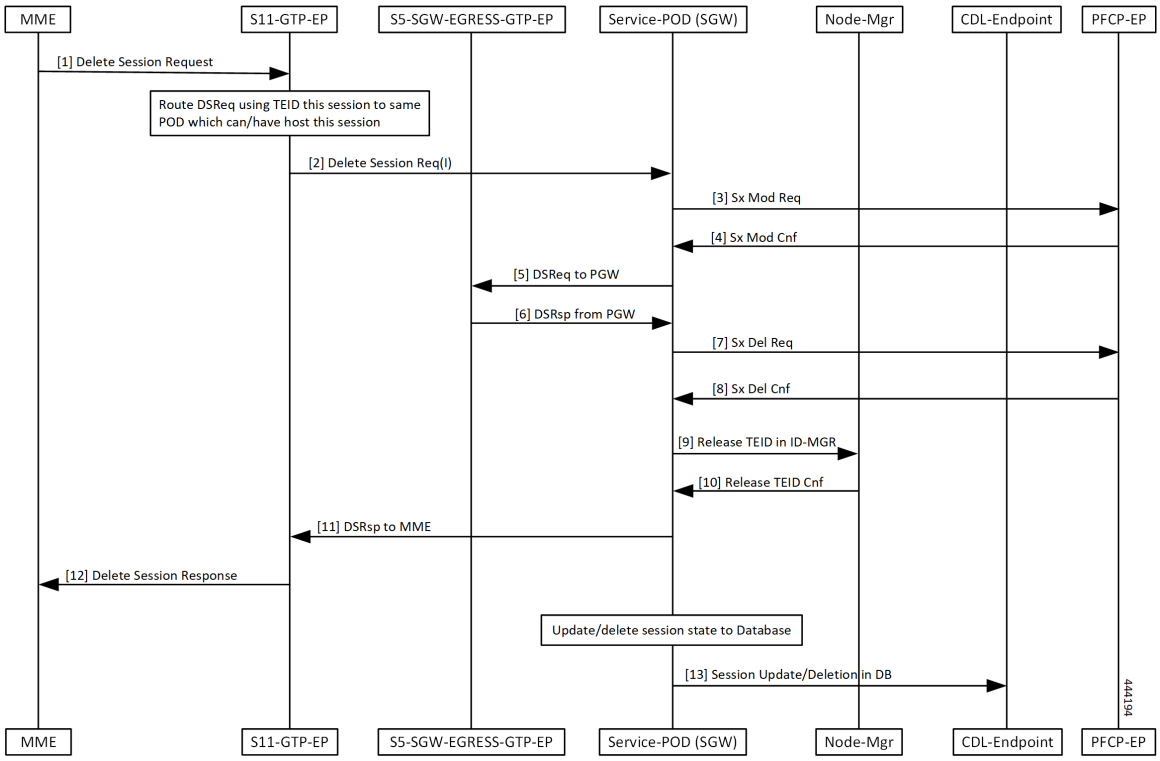


Table 50: Delete from MME Call Flow Description

Step	Description
1	The MME sends the Delete Session Request to the S11-GTP-EP.
2	The S11-GTP-EP routes this message with TEID value to the Service-POD (SGW) which handles this session.
3	The Service-POD (SGW) sends the Sx Modification Request to PFCP-EP.
4	The PFCP-EP sends the Sx Modification Confirmation to the Service-POD (SGW).
5	The Service-POD (SGW) sends the Delete Session Request to the PGW through the S5-SGW-EGRESS-GTP-EP.
6	The Service-POD (SGW) receives the Delete Session Request from the PGW through the S5-SGW-EGRESS-GTP-EP.
7	The Service-POD (SGW) sends the Sx Delete Request to PFCP-EP.
8	The Service-POD (SGW) receives the Sx Delete Confirmation from PFCP-EP.
9	The Service-POD (SGW) sends Release TEID in ID-MGR to Node-Mgr.
10	The Service-POD (SGW) receives the Release TEID Confirmation from the Node-Mgr.

Step	Description
11	The Service-POD (SGW) sends the Delete Session Response to S11-GTP-EP.
12	The S11-GTP-EP sends the Delete Session Response to the MME.
13	The Service-POD (SGW) sends the Session Update or Delete in database message to the CDL.

Figure 30: Delete from PGW Call Flow

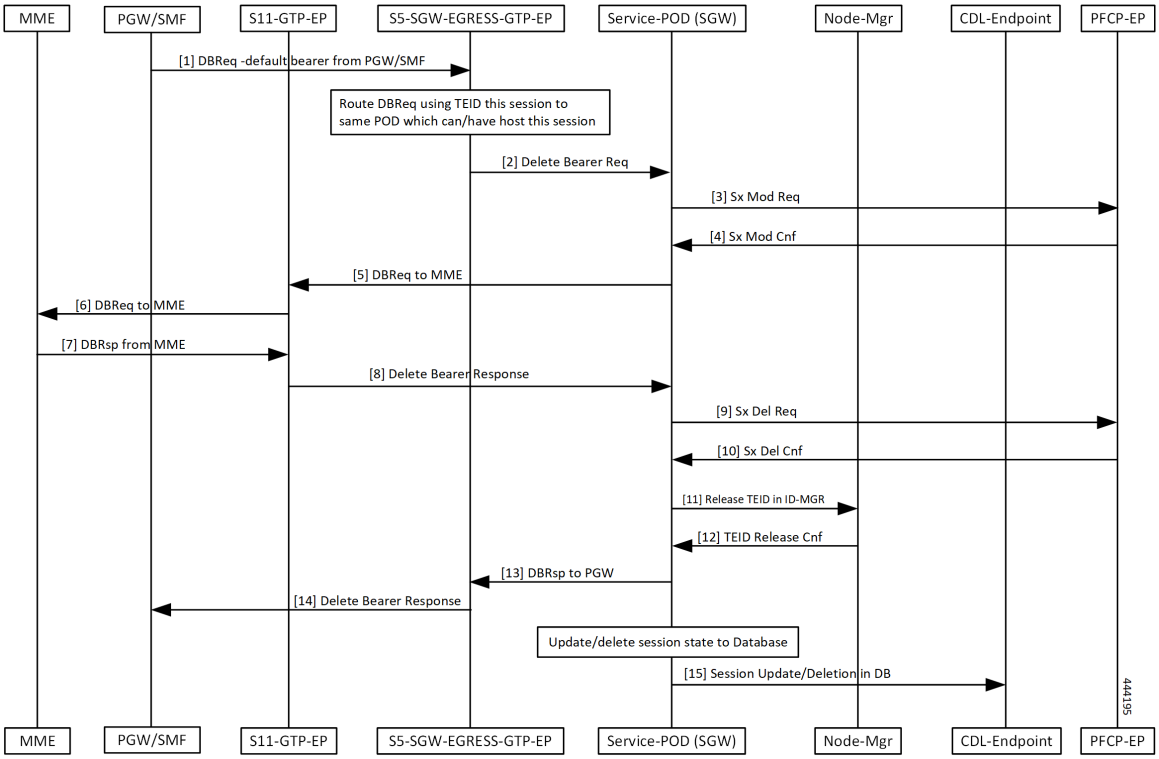
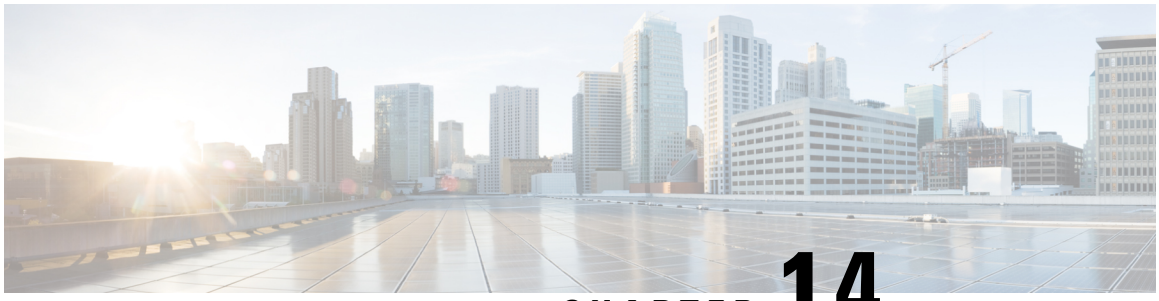


Table 51: Delete from PGW Call Flow Description

Step	Description
1	The PGW/SMF sends the Delete Bearer Request to the S5-SGW-EGRESS-GTP-EP.
2	The S5-SGW-EGRESS-GTP-EP performs routing of this message with TEID value to the same pod that has hosted this session. The S5-SGW-EGRESS-GTP-EP sends the Delete Bearer Request to the Service-POD (SGW).
3	The Service-POD (SGW) sends the Sx Modification Request to PFCP-EP and receives Sx Mod Cnf from it.
4	The PFCP-EP sends the Sx Modification Confirmation to the Service-POD (SGW).
5	The Service-POD (SGW) sends the Delete Bearer Request to the MME through the S11-GTP-EP.
6	The S11-GTP-EP forwards the Delete Bearer Request to the MME.

Step	Description
7	The MME sends the Delete Bearer Response to the S11-GTP-EP.
8	The S11-GTP-EP forwards the Delete Bearer Response to the Service-POD (SGW).
9	The Service-POD (SGW) sends the Sx Delete Request to the PFCP-EP.
10	The Service-POD (SGW) receives the Sx Delete Confirmation from the PFCP-EP.
11	The Service-POD (SGW) sends the Release TEID in ID-MGR to the Node-Mgr.
12	The Service-POD (SGW) receives the Release TEID Confirmation from the Node-Mgr.
13	The S11-GTP-EP sends the Delete Bearer Response to the PGW through S5-SGW-EGRESS-GTP-EP.
14	The S5-SGW-EGRESS-GTP-EP sends the Delete Bearer Response to the PGW/SMF.
15	The Service-POD (SGW) sends the Session Update or Delete in database message to the CDL.



CHAPTER 14

Downlink Data Notification

- Feature Summary and Revision History, on page 139
- Feature Description, on page 140
- DDN Message Handling, on page 140
- Control Messages Triggered DDN Support, on page 148
- DDN Advance Features, on page 150

Feature Summary and Revision History

Summary Data

Table 52: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	DDN Message Handling Support: Enabled - Always-on Control Messages Triggered DDN Support: Disabled - Configuration required to enable DDN Advance Features: Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 53: Revision History

Revision Details	Release
Enhancement introduced. Added support for DDN Advance Features.	2021.02.0

Revision Details	Release
First introduced.	2021.01.0

Feature Description

The following sub-features are associated with this feature:

- DDN Message Handling
- Control Messages Triggered DDN
- Downlink Data Notification Delay
- High Priority Downlink Data Notification
- DDN Throttling

DDN Message Handling

Feature Description

cnSGW-C supports handling of the Downlink Data Notification (DDN) functionality that includes:

- Generating a DDN message towards the MME to page the UE on arrival of downlink data when UE is in IDLE state.
- Handling DDN ACK/DDN failure indication.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Downlink Data Notification Success Call Flow

This section describes the Downlink Data Notification Success call flow.

Figure 31: Downlink Data Notification Success Procedure Call Flow

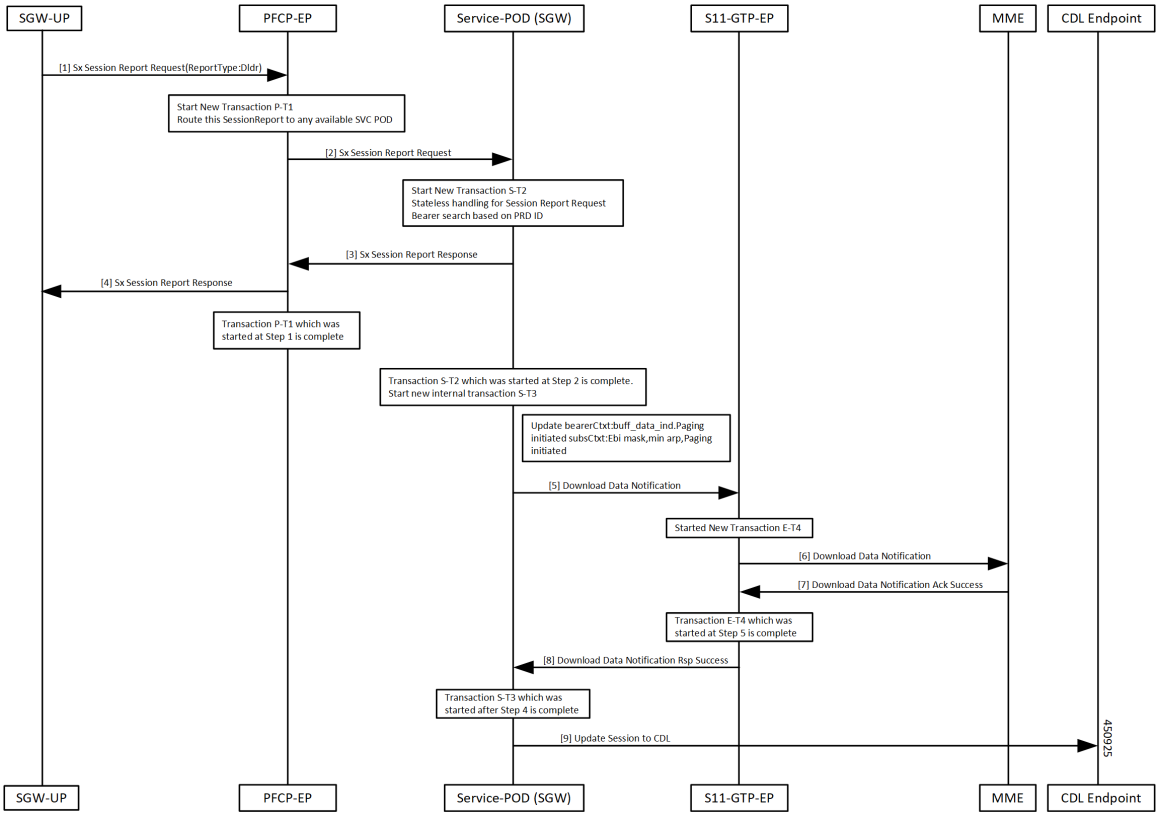


Table 54: Downlink Data Notification Success Procedure Call Flow Description

Step	Description
1	SGW-UP sends the Session Report Request to the PFCP-EP pod.
2	After receiving the Session Report Request, the PFCP-EP performs the following: • Starts a new P-T1 transaction. • Checks for the interface type. If its Sxa interface, it finds the available SGW-service pod and routes the request accordingly. • Sends the Sx Session Report Request to the SGW-service pod.

Step	Description
3	Upon reception of the Sx Session Report Request, SGW-service pod: • Creates a new S-T2 transaction. • Based on the message type received, updates the state processing not required for this message. • Handles the non-state processing transaction. (High priority is given to handle such messages). • Searches for the bearer based on PDR ID. If the bearer isn't found, the SGW-service pod fills the cause as **request rejected** in the Session Report Response. • If the received report type in the request isn't valid/supported, SGW-service pod fills the cause as **request rejected** and sends the Sx Session Report Response.
4	PFCP-EP forwards the Sx Session Report Response to the SGW-UP.
5	P-T1 transaction which is started at step one is completed. At SGW-service pod: • S-T2 transaction which is started at step two is completed. • If the Sx Session Report Response is success, a new internal transaction S-T3 is started with the same buffer as of the Session Report Request. • A DDN procedure for DLDR report type is initiated. • Bearer information is extracted from the received PDR ID. • Bearer context is updated with buffer-data_ind. • Initiated the DDN with EBI of bearers, which has downlink data, and minimum ARP among these bearers. • Sends the DDN to the S11-GTP-EP pod.
6	After receiving the DDN, S11-GTP-EP: • Creates a new E-T4 transaction. • Sends the DDN to the MME.
7	MME sends the DDN ACK Success to the S11-GTP-EP.
8	The transaction S-T3 which is started after step four is complete. S11-GTP-EP sends the DDN Response success to SGW-service pod.
9	SGW-service pod updates the CDL.

Downlink Data Notification Failure Call Flow

This section describes the Downlink Data Notification Failure call flow.

Figure 32: Downlink Data Notification Failure Call Flow

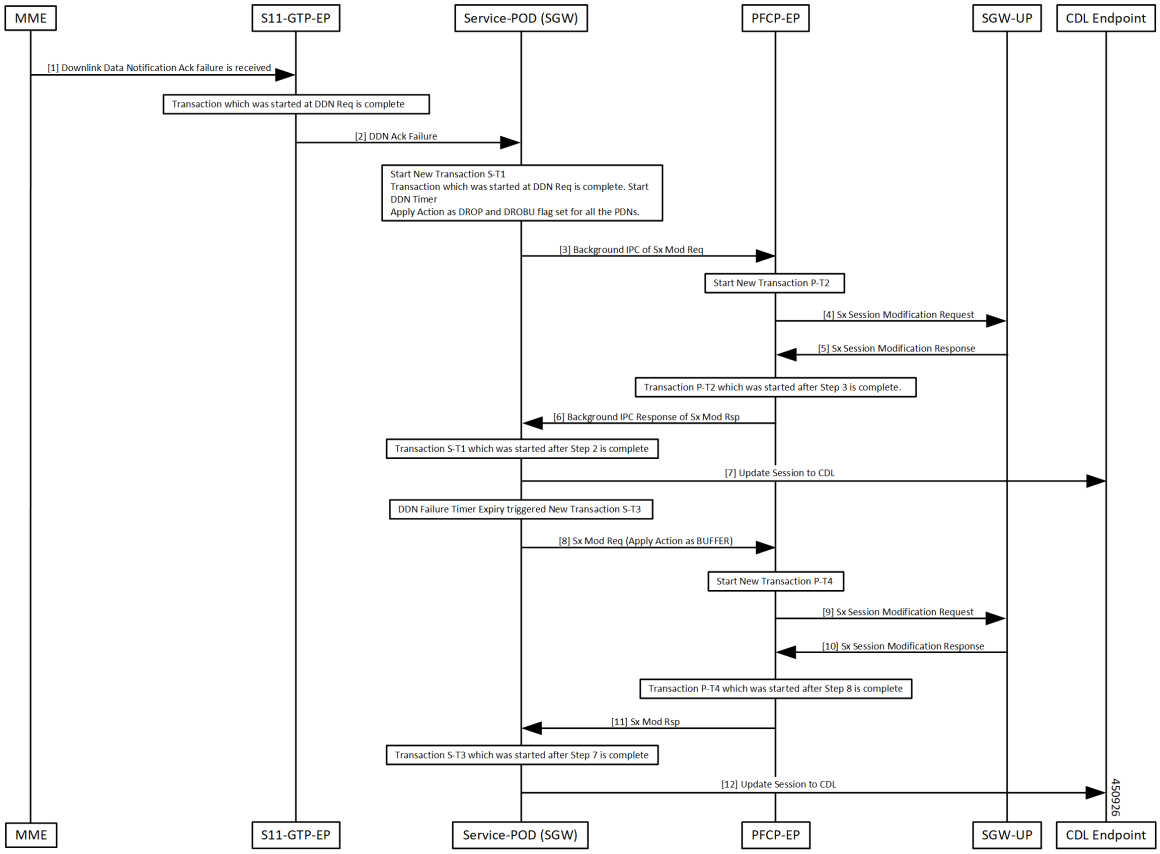


Table 55: Downlink Data Notification Failure Procedure Call Flow Description

Step	Description
1	S11-GTP-EP pod receives DDN ACK Failure.
2	The transaction started while sending the DDN Request ends. S11-GTP-EP forwards the DDN ACK Failure to the SGW-service pod.

Step	Description
3	After receiving the DDN ACK Failure at the SGW-service pod: • Decides the paging state based on the cause received: • EGTP_CAUSE_CONTEXT_NOT_FOUND: Submit internal transaction for call deletion. • EGTP_CAUSE_UNABLE_TO_PAGE_UE • EGTP_CAUSE_UNABLE_TO_PAGE_UE_DUE_TO_SUSPENSION • EGTP_CAUSE_UE_ALREADY_REATTACHED • EGTP_CAUSE_TEMP_REJECTED_DUE_TO_HANDOVER_IN_PROGRESS • Checks if the PDNs are in connected state to initiate the Sx Modify Request. Minimum one one PDN should be in the CONNECTED state. • Submits internal transactions to handle these paging failure causes. • Ends the current procedure and transaction. • In the new transaction of handling paging failures, derives all the PDNs for which you want to send Sx Modify request. • Based on the paging state, derives paging action and send Sx Modify Request based on the action required. Sends background IPC request for Sx Modification Request to PFCP-EP pod. Create a new transaction P-T2.
4	After receiving background IPC request for Sx Modification request, PFCP-EP: • Starts a new P-12 transaction. • Sends the o the SGW-UP.
5	PFCP-EP receives the Sx Modification Response from the SGW-UP.
6	The transaction P-T2 started at step three is complete. PFCP-EP pod sends background IPC response to the SGW-service pod.
7	The transaction S-T1 started at step two is complete. SGW-service pod updated the CDL with buff_data_ind at bearer level flag.
8	On DDN Failure timer expiry, a new transaction S-T3 is started. SGW-service pod sends background IPC request for the Sx Modification Request to the PFCP-EP pod with Apply Action as BUFFER.
9	A new P-T4 transaction is created. PFCP-EP pod sends the Sx Modification Request to the SGW-UP.
10	SGW-UP sends the Sx Modification Response to the PFCP-EP pod.

Step	Description
11	The transaction P-T4 started at step eight is complete. PFCP-EP pod forwards the Sx Modification Response to the SGW-service pod.
12	The transaction S-T3 started at step seven is complete. SGW-service pod updates the CDL.

No User Connect Retry Timer Call Flow

This section describes the No User Connect Retry Timer call flow.

Figure 33: No User Connect Retry Timer Call Flow

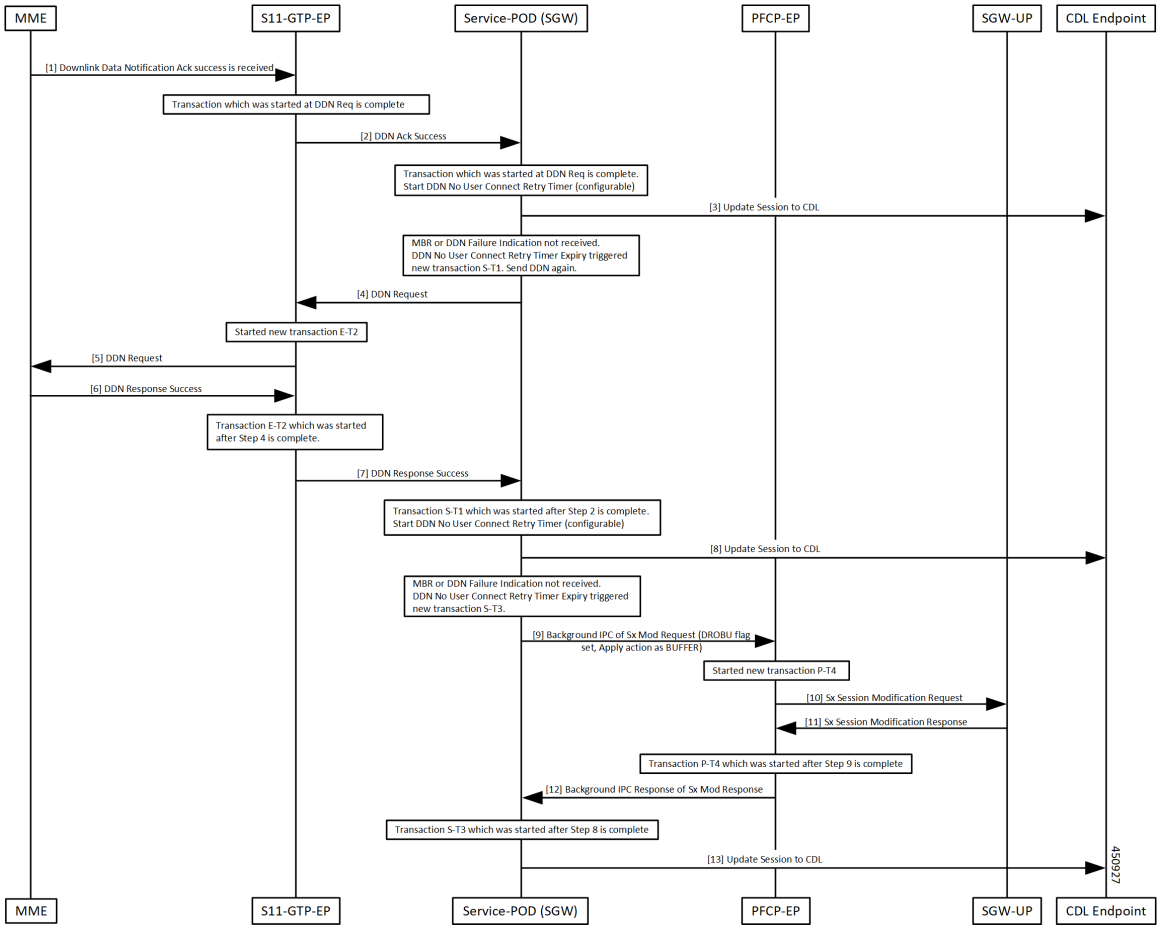


Table 56: No User Connect Retry Timer Call Flow Description

Step	Description
1	Received DDN ACK success at the S11-GTP-EP pod.

Step	Description
2	The transaction started while sending the DDN ends here. S11-GTP-EP sends the DDN ACK success to the SGW-service pod.
3	No User Connect Retry timer is started at the SGW-C pod. This timer is configurable. SGW-service pod updates the CDL.
4	SGW-service pod sends the DDN Request to the S11-GTP-EP pod, when: • The DDN Failure Indication/MBR is not received • No User Connect Retry timer expires. A new transaction S-T1 is created.
5	A new E-T2 transaction is created. S11-GTP-EP pod forwards the DDN Request to MME.
6	MME sends the DDN Response to the S11-GTP-EP.
7	The transaction E-T2 started at step four is complete. S11-GTP-EP forwards the DDN Response Success to the SGW-service pod.
8	S-T1 transaction started at step two is completed. No User Connect Retry timer is started at the SGW-C pod. This timer is configurable. SGW-service pod updates the CDL.
9	If DDN Failure Indication/MBR is not received, No User Connect Retry expiry triggered. A new transaction S-T3 is created. SGW-service pod sends the background IPC request for Sx Modification request to the PFCP-EP pod (DROBU flag and Apply Action as BUFFER).
10	A new transaction P-T4 is created. PFCP-EP pod sends the Sx Modification Request to the SGW-U pod.
11	PFCP-PE pod receives the Sx Modification Response.
12	The transaction P-T4 started at step nine is complete. PFCP-EP pod sends the background IPC response to the SGW-service pod.
13	The transaction S-T3 started at step eight is complete. CDL is updated.

Feature Configuration

Configuring this feature involves the following steps:

Configuring the DDN Failure Timer

DDN Failure Timer is configured under the `sgw-profile`.

To configure this feature, use the following configuration:

```
config
   profile sgw sgw_name
      ddn failure-action-drop-timer timer_value
      ddn timeout-purge-session { true | false }
      end
```

NOTES:

- **ddn failure-action-drop-timer** *timer_value*—Specify the duration of the DDN packet drop timer. During this specified timeframe, the DDN is not sent to the UE. This timer is used, when a notification of DDN ACK Failure or DDN Failure Indication is received. The default value is 300 seconds.

Note To disable the timer, set the timer value to zero.

- **ddn timeout-purge-session { true | false }**—Specify the option to enable or disable the DDN timeout purge session. The default value is false.

Configuration Example

The following is an example configuration.

```
config
profile sgw sgw1
ddn failure-action-drop-timer 60
ddn timeout-purge-session false
end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw
profile sgw sgw1
locality LOC1
fqdn 209.165.201.1
ddn failure-action-drop-timer 60
ddn timeout-purge-session false
end
```

Configuring DDN No User Connect Retry Timer

This section describes how to configure the DDN No User Connect Retry Timer.

DDN No User Connect Retry Timer can be configured under sgw-profile.

To configure this feature, use the following configuration:

```
config
  profile sgw sgw_name
   ddn no-user-connect-retry-timer timer_value
   end
```

NOTES:

- **ddn no-user-connect-retry-timer** *timer_value* - Specify the DDN retry timer used when DDN Ack is received with Success and MBR is not received. Default value is 60 seconds.

  To disable the timer, set the value to 0.

Configuration Example

The following is the sample configuration.

```
config
 profile sgw sgw1
  ddn no-user-connect-retry-timer 120
  end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw
profile sgw sgw1
locality LOC1
fqdn cisco.com.apn.epc.mnc456.mcc123
ddn failure-action-drop-timer 60
ddn no-user-connect-retry-timer 120
```

Control Messages Triggered DDN Support

Feature Description

This feature supports paging the UE for the PGW-initiated control procedures when the UE is in IDLE mode.

Note This feature is CLI controlled.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Downlink Data Notification for PGW-initiated procedure with Cloud Native Call Flow

This section describes the DDN for the PGW-initiated procedure with Cloud Native call flow.

Figure 34: Downlink Data Notification for PGW initiated Procedure with Cloud Native Call Flow

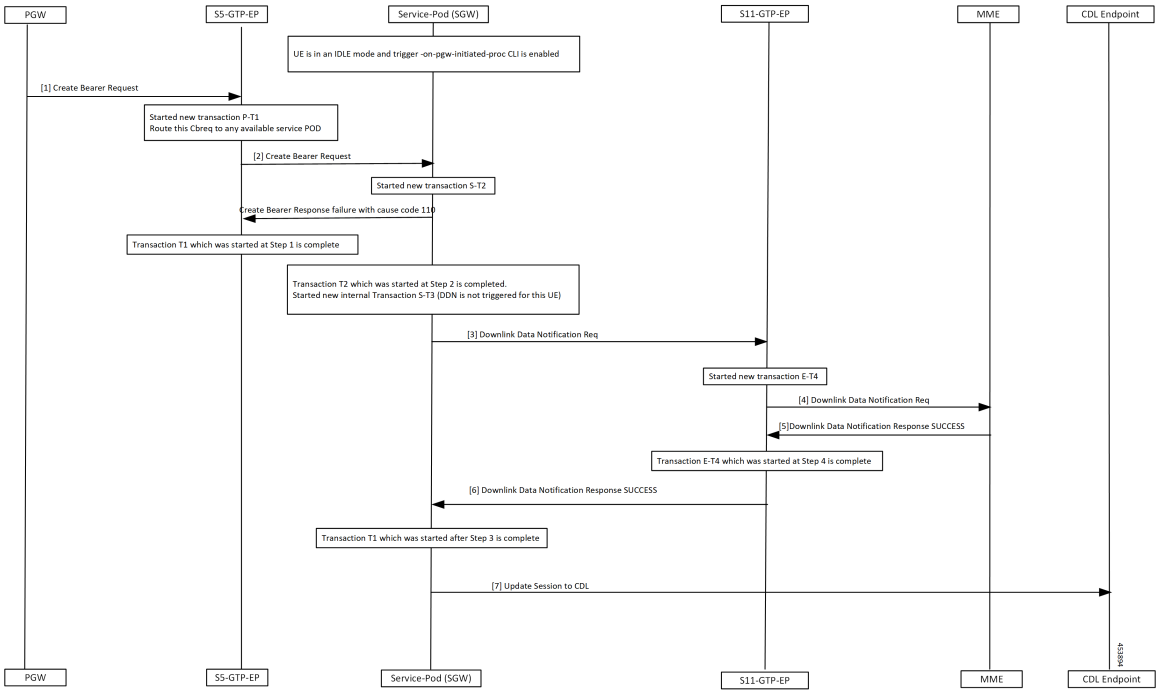


Table 57: Downlink Data Notification for PGW initiated procedure (CBR) with Cloud Native Call Flow Description

Step	Description
1, 2	Enabled trigger-on-pgw-initiated-proc CLI and state of the UE is in IDLE mode. S5-GTP-EP receives the CBR from the PGW and forwards it to the SGW-service pod. SGW-service pod starts a new S-T2 transaction. SGW-service pod sends failure response to the S5-GTP-EP with cause code 110.
3	The T2 transaction which started in step two is completed. A new S-T3 transaction is started for the UE for which DDN is not triggered. SGW-service pod initiates the DDN Request to the theS11-GTP-EP.
4	A new E-T4 transaction is started. S11-GTP-EP forwards the DDN Request to the MME.
5	S11-GTP-EP receives the DDN Response success from the MME.
6	Transaction E-T4 which started in step four is completed. S11-GTP-EP sends the DDN Response success to the SGW-service pod.
7	Transaction T1 which is started in step three is completed. SGW-service pod updates the session to CDL.

Feature Configuration

To configure this feature, use the following configuration:

```
config
 profile sgw sgw_name
  ddn trigger-on-pgw-initiated-proc
  end
```

NOTES:

- **ddn trigger-on-pgw-initiated-proc**—When UE is in IDLE mode, the DDN triggers paging for PGW-initiated procedures. SGW sends failure response to the PGW with cause code 110.

Configuration Example

The following is an example configuration.

```
config
 profile sgw sgw1
  ddn trigger-on-pgw-initiated-proc
  end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw
profile sgw sgw1
locality LOC1
fqdn 209.165.201.1
ddn failure-action-drop-timer 60
ddn no-user-connect-retry-timer 120
ddn trigger-on-pgw-initiated-proc
exit
```

Disabling the DDN Control Procedure

Use `no ddn trigger-on-pgw-initiated-proc` to disable DDN Control Procedure feature.

DDN Advance Features

Feature Description

This feature supports the following:

- Downlink Data Notification Delay
- High Priority Downlink Data Notification
- DDN Throttling

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

DDN Delay Call Flow

This section describes DDN Delay call flow.

Figure 35: DDN Delay Call Flow

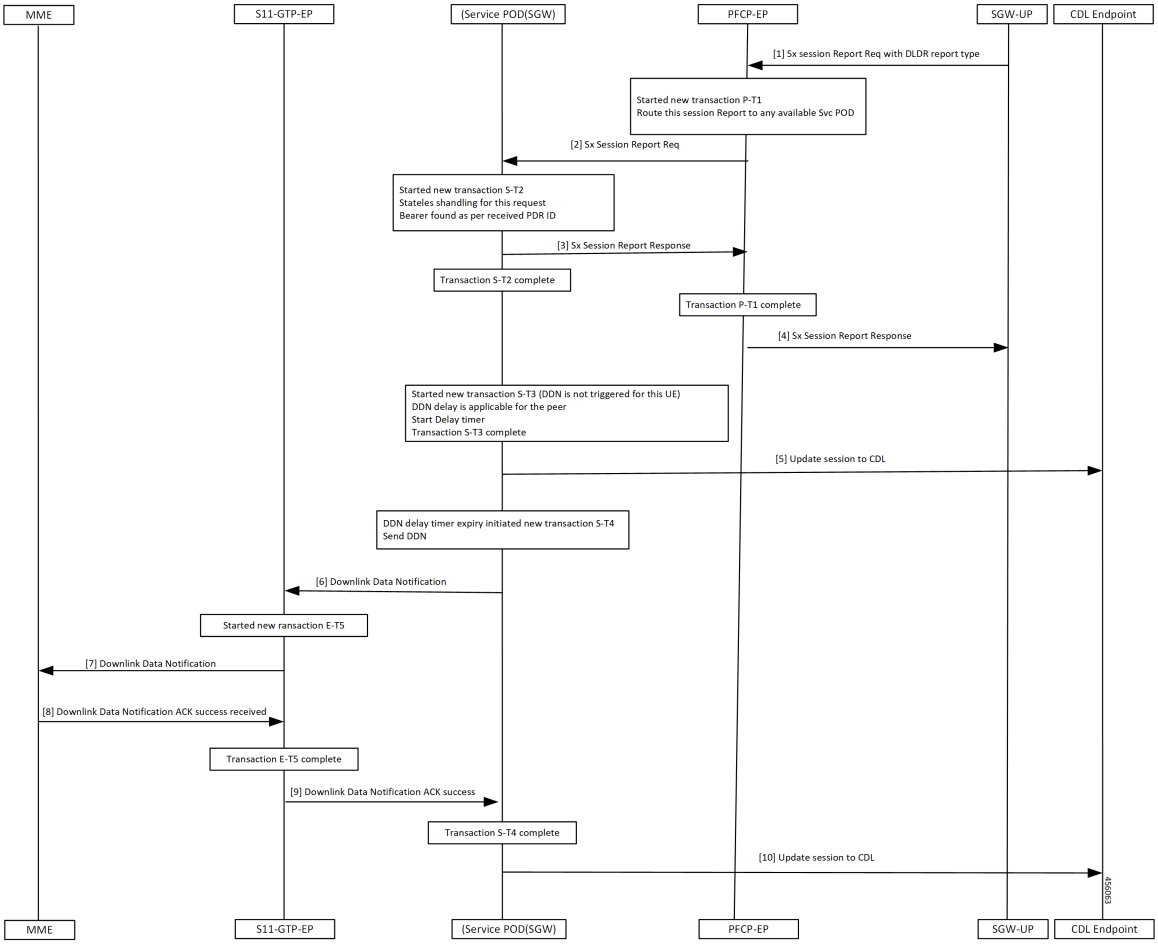


Table 58: DDN Delay Call Flow Description

Step	Description
1	Received downlink data when UE is in IDLE state. SGW-UP sends the Sx Report Request with report type as DLDR with corresponding PDR ID to the PFCP-EP.

Step	Description
2	Started a new P-T1 transaction. PFCP-EP pod: • Checks the available service pod. • Sends the Sx Session Report to the the SGW-service pod.
3	A new transaction S-T2 is stared. SGW-CP sends success response to the SGW-UP, when a bearer found at CP for this PDR-ID.
4	The S-T2, P-T1 transactions are completed. PFCP-EP sends the Sx Session Report Response to the SGW-UP.
5	A new transaction S-T3 is started when DDN is not triggered for this UE. Sgw-service pod gets the peer information to check if the peer configured with the DDN delay value. DDN delay timer is triggered, if DDN delay configured. S-T3 transaction is completed. SGW-service pod sends the CDL update.
6, 7	A new S-T4 transaction started. SGW-service pod sends the DDN to the S11-GTP-EP. A new E-T5 transaction is started. S11-GTP-EP forwards the DDN to the MME.
8, 9	MME sends the DDN ACK success to the S11-GTP-EP. Transaction E-T5 started in step seven is completed. S11-GTP-EP forwards the DDN ACK success towards the SGW-service pod.
10	Transaction S-T4 started in step six is completed. SGW-service pod updates session information to CDL.

High Priority DDN Call Flow

This section describes High Priority DDN call flow.

Figure 36: High Priority DDN Call Flow

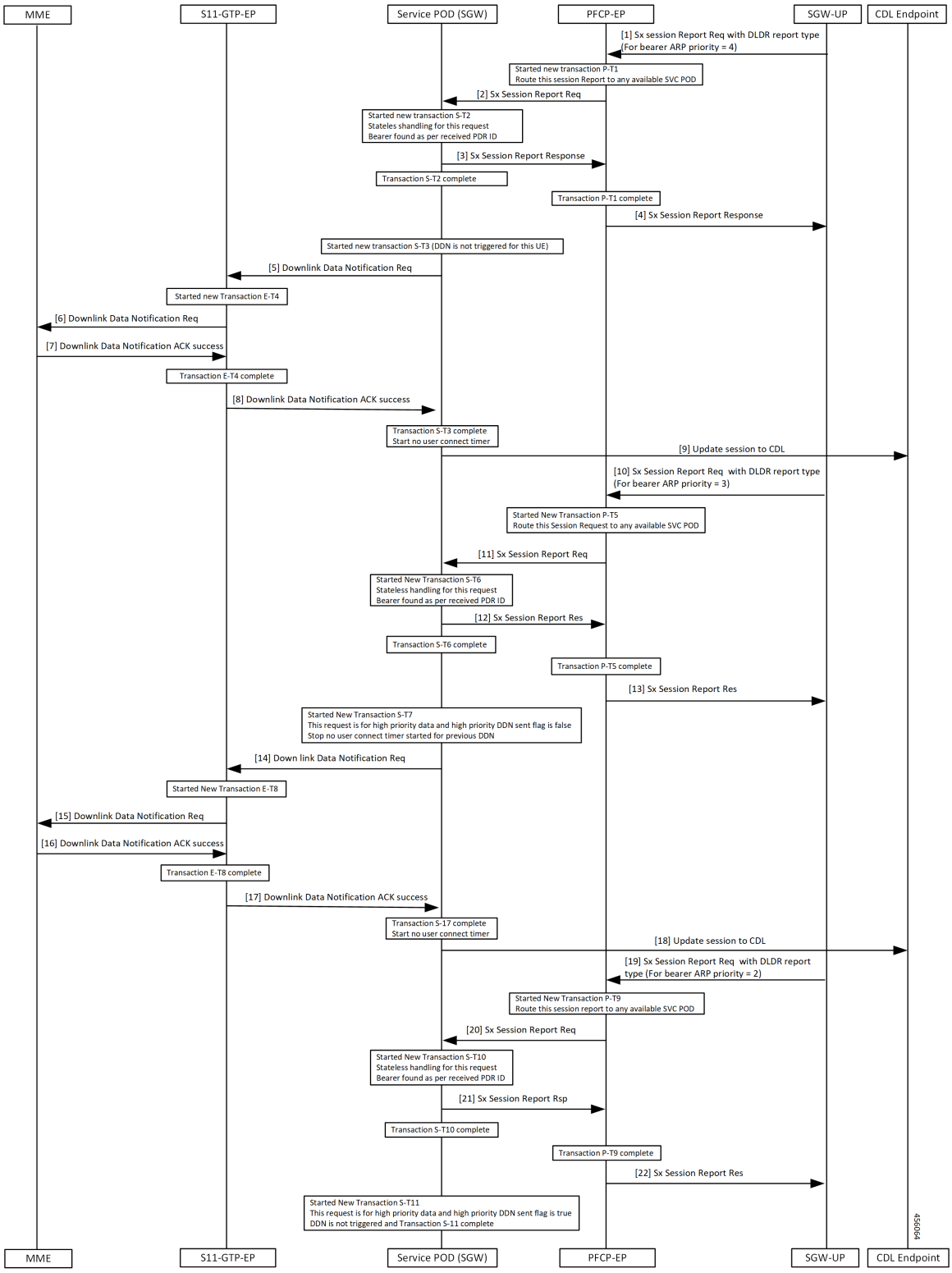


Table 59: High Priority DDN Call Flow Description

Step	Description
1	Bearer received downlink data with ARP priority value as four, when UE is in IDLE state. SGW-UP sends the Sx Report Request to the PFCP-EP with report type as DLDR with corresponding PDR ID.
2	New P-T1 transaction is started and routed the session report to all available service pods. PFCP-EP sends the Sx Session Report Request to the SGW-service pod.
3	New S-T2 transaction is started and the SGW-service pod sends Sx Session Report Response to the PFCP-EP.
4	Transaction P-T1 and S-T2 completed and the PFCP-EP forwards the Sx Session Report Response to the SGW-UP.
5	New S-T3 transaction is started for which the DDN isn't triggered. SGW-service pod sends the DDN Request to the S11-GTP-EP.
6	New E-T4 transaction is started and the S11-GTP-EP forwards the DDN Request to the MME.
7	MME sends the DDN ACK success to the S11-GTP-EP.
8	Transaction E-T4 is completed. S11-GTP-EP forwards the DDN ACK success to the SGW-service pod.
9	Transaction S-T3 completed. SGW-service pod triggers No User Connect timer and updates session to CDL.
10	SGW-UP sends the Sx Session Report Request to the PFCP-EP with report type as DLDR for bearer whose ARP priority value is three.
11	New P-T5 transaction is started and routed the session report to all the available service pods. PFCP-EP sends the Sx Session Report Request to the SGW-service pod.
12	New transaction S-T6 started SGW-service pod sends the Sx Session Report Response to the PFCP-EP when bearer found as per the received PDR ID.
13	Transaction S-T6 and P-T5 completed and PFCP-EP forwards the Sx Session Report Response to the SGW-UP.
14	New transaction S-T7 started and data, high priority DDN sent with the flag value as False. No User Connect timer is topped. SGW-service pod sends the DDN Request to the S11-GTP-EP.
15	New E-T8 transaction is started and the S11-GTP-EP forwards the DDN Request to the MME.
16	MME sends the DDN ACK success to the S11-GTP-EP.

Step	Description
17	S11-GTP-EP forwards the DDN ACK success to the SGW-service pod for this PDR ID.
18	Transaction S-17 completed. SGW-service pod triggers the No User Connect timer when received DDN ACK success and updated the session to CDL.
19	Bearer received the downlink data with ARP priority value as two. SGW-UP sends the Sx Report Request to the PFCP-EP with report type as DLDR with corresponding PDR ID.
20	New transaction P-T9 started and routed the session report to all the available service pods. PFCP-EP sends the Sx Session Report Request to the SGW-service pod.
21	New transaction S-T10 started and the SGW-service pod sends the Sx Session Report Response to the PFCP-EP when the bearer found as per the received PDR ID.
22	Transaction S-T10 and P-T9 completed and the PFCP-EP forwards the Sx Session Report Response to the SGW-UP. At SGW-service pod: • New transaction S-T11 started and data, high priority DDN sent with the flag value as True. SGW-service pod stops No User Connect timer. • SGW-service pod doesn't trigger DDN when high priority DDN already initiated. Transaction S-11 is completed.

DDN Throttling Call Flow

This section describes DDN Throttling call flow.

Figure 37: DDN Throttling Call Flow

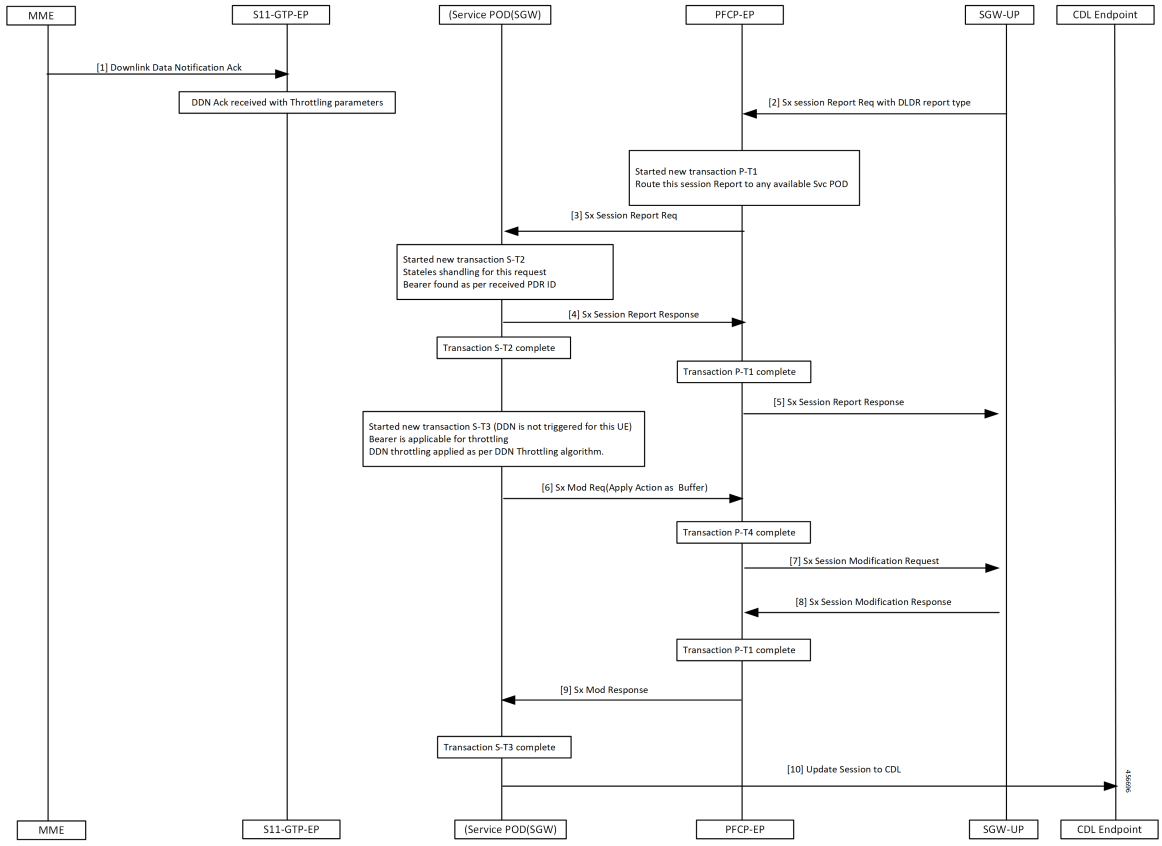


Table 60: DDN Throttling Call Flow Description

Step	Description
1	Received DDN with throttling parameters when UE is in IDLE state. MME sends the DDN ACK to the S11-GTP-EP.
2	SGW-UP sends the Sx Report Request with report type as DLDR with corresponding PDR ID to PFCP-EP.
3	PFCP-EP triggers a new P-T1 transaction and routes the Sx Report Request to available service pod. PFCP-EP sends the Sx Session Report Request to the SGW-service pod.
4	Started a new S-T2 transaction. Get peer information to check if DDN Throttle is active for this peer. Check if priority of this bearer is more than the configured ARP watermark SGW-service pod sends the Sx session Report Response to the PFCP-EP.

Step	Description
5	S-T2 transaction started in step four is completed. P-T1 transaction started in step threee is completed. When a bearer found at CP for this PDR ID, the PFCP-EP sends success response to the SGW-UP.
6	A new S-T3 transaction is started for the UE for which DDN is not triggered. Apply DDN algorithm to check if the DDN must be throttled. If DDN throttled, SGW-service pod sends the Sx Modification Request with Apply Action as BUFFER towards PFCP-EP.
7, 8	P-T4 transaction is completed. PFCP-EP sends the Sx Session Modification Request to the SGW-UP and receives the Sx Session Modification Response from the SGW-UP.
9	P-T1 transaction started in step five is completed. PFCP-EP sends Sx Modification Response to the SGW-service pod.
10	S-T3 transaction started in step six is completed. SGW-service pod updates the session to CDL.

Note cnSGW does not throttle outgoing downlink data notification messages for emergency or WPS sessions.

Standards Compliance

The Downlink Data Notification Support feature complies with the following standards:

- *3GPP TS 23.401, "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access"*
- *3GPP TS 23.402, "Architecture enhancements for non-3GPP accesses"*
- *3GPP TS 29.274, "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3"*
- *3GPP TS 23.214, "Architecture enhancements for control and user plane separation of EPC nodes"*
- *3GPP TS 29.244, "Interface between the Control Plane and the User Plane nodes"*
- *3GPP TS 24.008, "Mobile radio interface Layer 3 specification; Core network protocols; Stage 3"*

Feature Configuration

By default, the DDN throttling is always enabled.

Note cnSGW-C handles DDN throttling parameters sent from the MME.

To configure this feature, use the following configuration:

```
config
 profile sgw sgw_name
  ddn throttle-arp-watermark arp_value
  end
```

NOTES:

- **ddn throttle-arp-watermark***arp_ value*—Specify the lowest priority ARP for DDN throttle.

  Throttling is applicable only for bearer having ARP PL value greater than the configured *value*. Must be an integer in the range of 0-15.

  By default, throttling is applicable for all bearers.

Configuration Example

The following is an example configuration.

```
config
 profile sgw sgw1
  ddn throttle-arp-watermark 3
  end
```

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

The following statistics are supported for the DDN Advance feature.

```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",
ddn_stats_type="control_proc_triggered",instance_id="0",service_name="sgw-service"}2
```

```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",ddn_stats_type="data_triggered",
instance_id="0",service_name="sgw-service"} 18
```

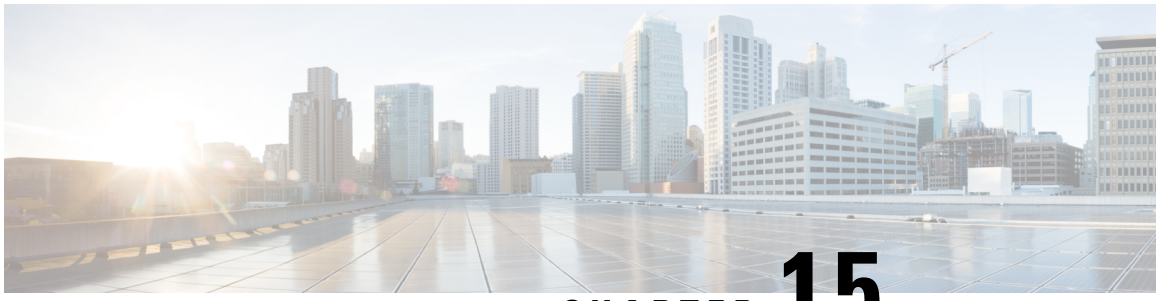
```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",ddn_stats_type="delayed",
instance_id="0",service_name="sgw-service"} 7
```

```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",ddn_stats_type="high_priority_initiated",
instance_id="0",service_name="sgw-service"} 3
```

```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",ddn_stats_type="high_priority_suppressed",
instance_id="0",service_name="sgw-service"} 1
```

```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",ddn_stats_type="throttled",
instance_id="0",service_name="sgw-service"} 6
```

- high_priority_initiated - DDN initiated count, due to high priority paging trigger.
- high_priority_suppressed - DDN high priority count which is suppressed. When a UE is already working on the high priority DDN-initiated paging request. It suppresses the incoming high priority paging request.
- throttled - DDN throttled count.
- delayed - DDN initiated count after the DDN delay timer.
- control_proc_triggered - The received count of paging triggers from control procedure when UE is in IDLE state.
- data_triggered - The received count of paging triggers from UPF for downlink data when UE is in IDLE state.



CHAPTER 15

DSCP Marking Support

- Feature Summary and Revision History, on page 161
- Feature Description, on page 162
- DSCP Marking for Data Packets, on page 162
- DSCP Marking for CP Signaling Messages, on page 164

Feature Summary and Revision History

Summary Data

Table 61: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	DSCP Marking for Data packets: Disabled – Configuration required to enable DSCP Marking for CP Signaling Messages: Enabled – Always-on
Related Documentation	Not Applicable

Revision History

Table 62: Revision History

Revision Details	Release
Validated support for Extended and Non-Standard (Operator-defined) QCI Values.	2021.02.3
Added support for DSCP Marking for CP Signaling Messages.	2021.02.0

Revision Details	Release
First introduced.	2021.01.0

Feature Description

Differentiated Services Code Point (DSCP) is a means of classifying and managing network traffic. It provides quality of service (QoS) in modern Layer 3 IP networks.

This feature supports the following:

- DSCP Marking for Data Packets
- DSCP Marking for CP Signaling Messages

DSCP Marking for Data Packets

Feature Description

This feature supports marking of DSCP with the combination of QCI and ARP.

It also supports the programming of the DSCP marking value to the User Plane (UP) for data packets.

How it Works

This section describes how this feature works.

DSCP Marking IEs

DSCP marking IEs are sent in the Sx Establishment Request or the Sx Modification Request message. These IEs are a part of Forwarding Action Rule (FAR) IE. The following are the supported IEs and their functions:

- Inner Packet Marking (Private Extension IE): Sends the user-datagram DSCP marking values to the UP.
- Transport Packet Marking (3GPP Spec-defined IE): Sends the encaps-header DSCP values to the UP.
- Transport Packet Marking Options (Private Extension IE): Sends copy-inner and copy-outer options of encaps-header marking to the UP.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   profile
      sgw-qos-profile qos_profile_name
         dscp-map
            operator-defined-qci non_standard_qos_class_id
```

```
qci qci_value
    downlink downlink_value
        user-datagram
            dscp-marking dscp_marking_value
        encaps-header
            dscp-marking dscp_marking_value
        encsp-header enscp_header_value
            dscp-marking dscp_marking_value
    uplink uplink_value
        user-datagram
            dscp-marking dscp_marking_value
        encaps-header
            dscp-marking dscp_marking_value
        encsp-header enscp_header_value
arp-priority-level arp_priority_level_value
    uplink
        user-datagram
            dscp-marking dscp_marking_value
        encaps-header
            dscp-marking dscp_marking_value
    downlink
        user-datagram
            dscp-marking dscp_marking_value
        encaps-header
            dscp-marking dscp_marking_value
            end
```

NOTES:

- **sgw-qos-profile** *qos_profile_name*—Specify the QoS profile configuration name for SGW.
- **dscp-map**—Configures QCI to DSCP-Marking mapping.
- **operator-defined-qci** *non_standard_qos_class_id*—Specify the non-standard QoS class identifier. Must be an integer in the range of 128-254.
- **qci** *qci_value*—Specify the standard QCI value. Must be an integer from the following options: 1-9, 65, 66, 69, 70, 80, 82, 83.
- **arp-priority-level** *arp_priority_value*—Specify the ARP Priority Level. Must be an integer in the range of 1-15.
- **uplink** *uplink_value*—Specify the uplink QCI value.
- **downlink** *downlink_value*—Specify the downlink QCI value.
- **gbr**—Specify the type of the QCI to GBR.
- **non-gbr**—Specify the type of the QCI to non-GBR.
- **encaps-header**—Specify the DSCP value to be applied to the encaps header.
- **user-datagram**—Specify the DSCP value to be applied to the user datagram.
- **copy-inner**—Starts copying the inner DSCP to outer value.
- **copy-outer**—Starts copying the outer DSCP to inner value.

- **dscp-marking** *dscp_marking_value*—Specify the DSCP value to be applied to packets. (A hexadecimal string value, starting with 0x. For example: 0x3F)
- **qci**—The QCI uplink and downlink options are the same. Similarly, the commands for **operator-defined-qci** and standard QCI are the same, the only difference is the mandatory selection of *bearer-type* in **operator-defined-qci**. You can also specify ARP along with the type of the bearer.

Configuration Example

The following is an example configuration.

```
config
  profile sgw-qos-profile q
    dscp-map qci 1 uplink encaps-header copy-inner user-datagram dscp-marking 0x1
    dscp-map qci 1 downlink user-datagram dscp-marking 0x2 encaps-header dscp-marking 0x3
    dscp-map qci 2 gbr uplink user-datagram dscp-marking 0x5 encaps-header dscp-marking 0x6

    dscp-map operator-defined-qci 128 gbr arp-priority-level 1 uplink user-datagram
dscp-marking 0x7
    end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw-qos-profile q
  profile sgw-qos-profile q
    dscp-map qci 1 uplink encaps-header copy-inner user-datagram dscp-marking 0x1
    dscp-map qci 1 downlink user-datagram dscp-marking 0x2 encaps-header dscp-marking 0x3
    dscp-map qci 2 gbr uplink user-datagram dscp-marking 0x5 encaps-header dscp-marking 0x6

    dscp-map operator-defined-qci 128 gbr arp-priority-level 1 uplink user-datagram
dscp-marking 0x7
    end
```

DSCP Marking for CP Signaling Messages

Feature Description

This feature supports the marking of DSCP values to control packets as per the configuration at the following interfaces:

- GTPC: S11, S5
- PFCP: Sxa

Feature Configuration

Configuring this feature involves the following steps:

- Configuring DSCP under the S11 Interface for the GTP Endpoint. For more information, refer to Configuring DSCP under S11 Interface for GTP Endpoint, on page 165.

- Configuring DSCP under the S5e Interface for the GTP Endpoint. For more information, refer to Configuring DSCP under S5e Interface for GTP Endpoint, on page 165.
- Configuring DSCP under the Sxa Interface for the Protocol Endpoint. For more information, refer to Configuring DSCP under Sxa Interface for Protocol Endpoint, on page 166.

Configuring DSCP under S11 Interface for GTP Endpoint

To configure this feature, use the following configuration:

```
config
    instance instance-id instance_id
        endpoint endpoint_name
        interface interface_name
        dscp dscp_value
        end
```

NOTES:

- **endpoint** *endpoint_name*—Specify the endpoint name.
- **interface** *interface_name*—Specify the endpoint interface name.
- **dscp** *dscp_value*—Specify the DSCP value. Must be a hexadecimal string starting with 0x (for example, 0x3F), or a decimal value (for example, 12). The decimal value must be in the range of 0-63.

Configuration Example

The following is an example configuration.

```
config
    instance instance-id 1
        endpoint gtp
        interface s11
        dscp 0x2
        end
```

Configuration Verification

To verify the configuration:

```
show running-config instance instance-id 1 endpoint
    endpoint gtp
    interface s11
    dscp 0x2
    end
```

Configuring DSCP under S5e Interface for GTP Endpoint

To configure this feature, use the following configuration:

```
config
    instance instance-id instance_id
        endpoint endpoint_name
        interface interface_name
        dscp dscp_value
        end
```

Configuration Example

The following is an example configuration.

```
config
    instance instance-id 1
        endpoint gtp
        interface s5e
        dscp 0x2
        end
```

Configuration Verification

To verify the configuration:

```
show running-config instance instance-id 1 endpoint
    endpoint gtp
    interface s5e
    dscp 0x2
    end
```

Configuring DSCP under Sxa Interface for Protocol Endpoint

To configure this feature, use the following configuration:

```
config
    instance instance-id instance_id
        endpoint endpoint_name
        interface interface_name
        dscp dscp_value
        end
```

Configuration Example

The following is an example configuration.

```
config
    instance instance-id 1
        endpoint gtp
        interface sxa
        dscp 0x2
        end
```

Configuration Verification

To verify the configuration:

```
show running-config instance instance-id 1 endpoint
    endpoint gtp
    interface sxa
    dscp 0x2
    end
```

Removing DSCP Configuration

When you remove the DSCP signaling configuration from the interface or endpoint, it uses the default marking. The default value is 10 or 0xa (in Hexadecimal).

To clear the DSCP configuration:

```
config
    instance instance-id instance_id
```

```
endpoint endpoint_name
interface interface_name
no dscp
end
```

Configuration Example

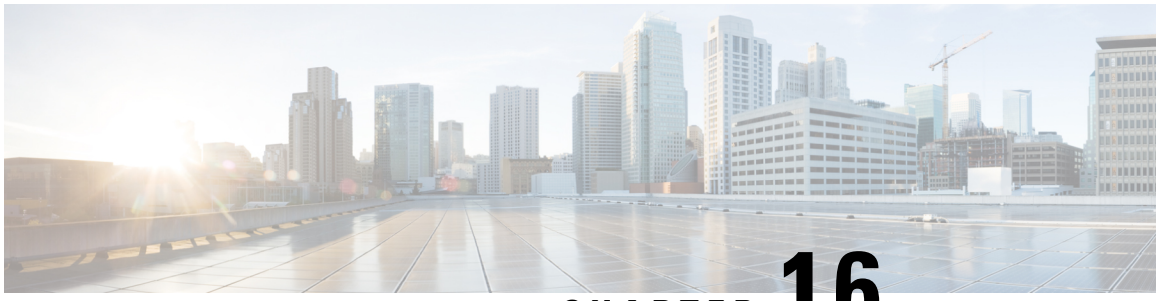
The following is an example configuration for the removal of the DSCP configuration.

```
config
    instance instance-id 1
        endpoint gtp
        interface s11
        no dscp
        end
```

Configuration Verification

To verify the DSCP configuration removal:

```
show running-config instance instance-id 1 endpoint
    instance instance-id 1
    endpoint gtp
    interface s5e
    dscp 0x4
    exit
    interface s11
    exit
    exit
    endpoint protocol
    interface sxa
    dscp 8
    end
```



CHAPTER 16

Dynamic Routing by Using BGP

- Feature Summary and Revision History, on page 169
- Feature Description, on page 170
- How it Works, on page 170
- Configuring Dynamic Routing Using BGP, on page 177
- Monitoring and Troubleshooting, on page 180

Feature Summary and Revision History

Summary Data

Table 63: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Default Setting	Disabled – Configuration required to enable
Related Changes in this Release	Not Applicable
Related Documentation	Not Applicable

Revision History

Table 64: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

Border Gateway Protocol (BGP) allows you to create loop-free inter-domain routing between autonomous systems (AS). An AS is a set of routers under a single technical administration. The routers can use an Exterior Gateway Protocol to route packets outside the AS. The Dynamic Routing by Using BGP feature enables you to configure the next-hop attribute of a BGP router with alternate local addresses to service IP addresses with priority and routes. The SMF BGP speaker pods enable dynamic routing of traffic by using BGP to advertise pod routes to the service VIP.

This feature supports the following functionality:

- Dynamic routing by using BGP to advertise service IP addresses for the incoming traffic.
- Learn route for outgoing traffic.
- Handling a BGP pod failover.
- Handling a protocol pod failover.
- Statistics and KPIs for the BGP speakers.
- Log messages for debugging the BGP speakers.
- Enable or disable the BGP speaker pods.
- New CLI commands to configure BGP.

How it Works

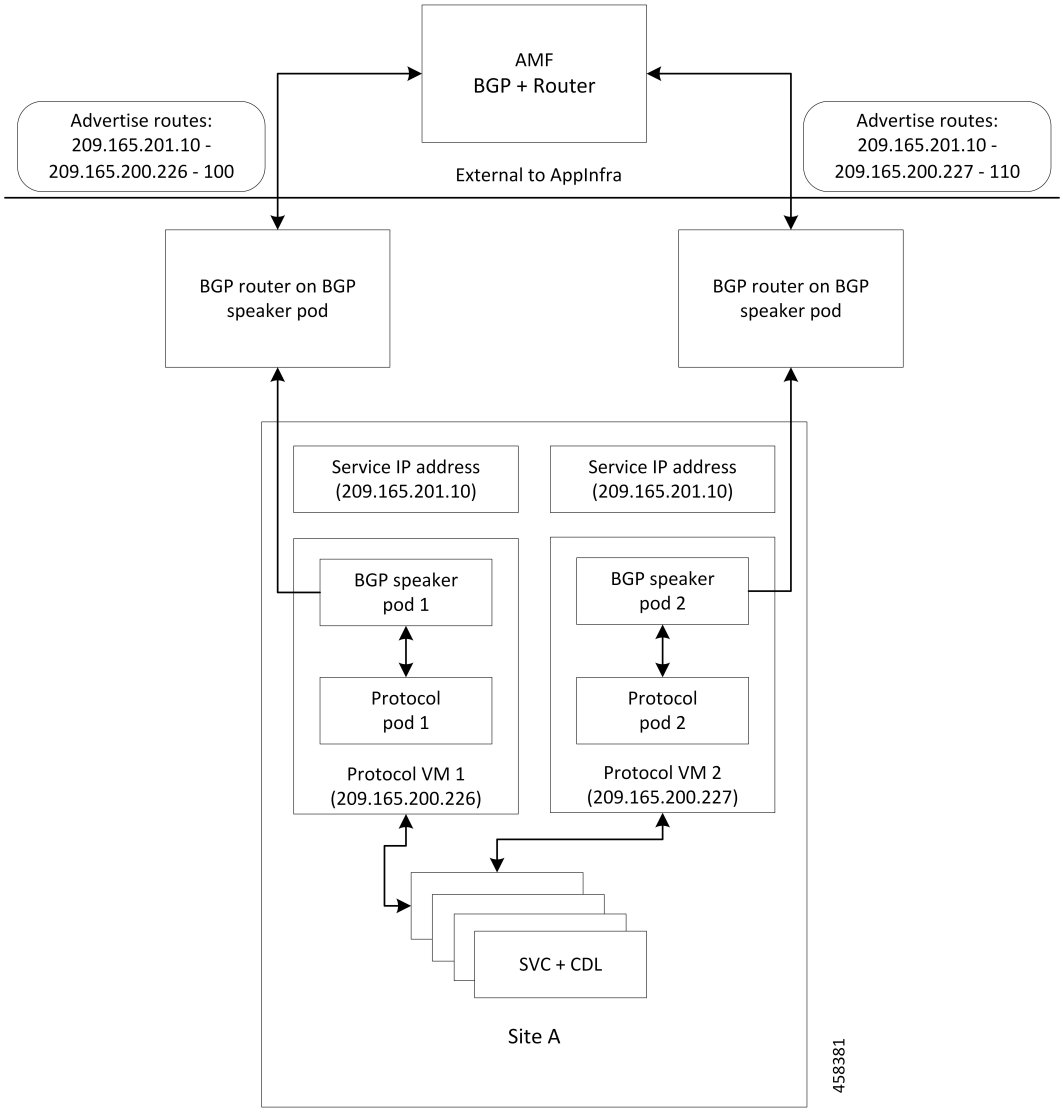
This section describes the operation of the Dynamic Routing feature.

Incoming Traffic

BGP uses TCP as the transport protocol, on port 179. Two BGP routers form a TCP connection between one another. These routers are peer routers. The peer routers exchange messages to open and confirm the connection parameters.

The BGP speaker publishes routing information of the protocol pod for incoming traffic in the active/standby mode. Use the following image as an example to understand the dynamic routing functionality. There are two protocol pods, pod1 and pod2. Pod1 is active and pod2 is in the standby mode. The service IP address, 209.165.201.10 is configured on both the nodes, 209.165.200.226 and 209.165.200.227. Pod1 is running on host 209.165.200.226 and pod2 on host 209.165.200.227. The host IP address exposes the pod services. BGP speaker publishes the route 209.165.201.10 through 209.165.200.226 and 209.165.200.227. It also publishes the preference values, 110 and 100 to determine the priority of pods.

Figure 38: Dynamic Routing for Incoming Traffic in the Active-standby Topology



For high availability, each cluster has two BGP speaker pods with active/standby topology. Kernel route modification is done at host/network level where the protocol pod runs.

MED Value

The Local Preference is used only for IGP neighbors, whereas the MED Attribute is used only for EGP neighbors. A lower MED value is the preferred choice for BGP.

Table 65: MED Value

Bonding Interface Active	VIP Present	MED Value	Local Preference
Yes	Yes	1210	2220
Yes	No	1220	2210

Bonding Interface Active	VIP Present	MED Value	Local Preference
No	Yes	1215	2215
No	No	1225	2205

Bootstrap of BGP Speaker Pods

The following sequence of steps set up the BGP speaker pods:

1. The BGP speaker pods use TCP as the transport protocol, on port 179. These pods use the AS number that is configured in the Ops Center CLI.
2. Register the Topology manager.
3. Select the Leader pod. The active speaker pod is the default choice.
4. Establish connection to all the BGP peers provided by the Ops Center CLI.
5. Publish all existing routes from ETCD.
6. Configure import policies for routing by using CLI configuration.
7. Start gRPC stream server on both the speaker pods.
8. Similar to the cache pod, two BGP speaker pods must run on each Namespace.

External Network Failure

The NF instance start-up causes the BGP Speaker K8s pod to configure the next-hop attribute of the BGP router with alternate local addresses to service IP addresses with priority and routes.

After the Geo HA is triggered, the path selection is based on the destination service IP address, path connectivity and the priority value.

Note The subscriber sessions are not impacted because of the transparent migration between pods.

Geo Switchover

The achieves geo switchover by transparently migrating service IP address to mated peer K8s cluster, rack collocated, or geo-located. During the NF start-up, all the K8s cluster Namespaces register with the next-hop BGP router to advertise its service IP address and local IP address along with the priority and route modifier values.

Each logical NF exposes separate NF instance toward NRF or DNS, separate configuration, and separate LCM for a Namespace.

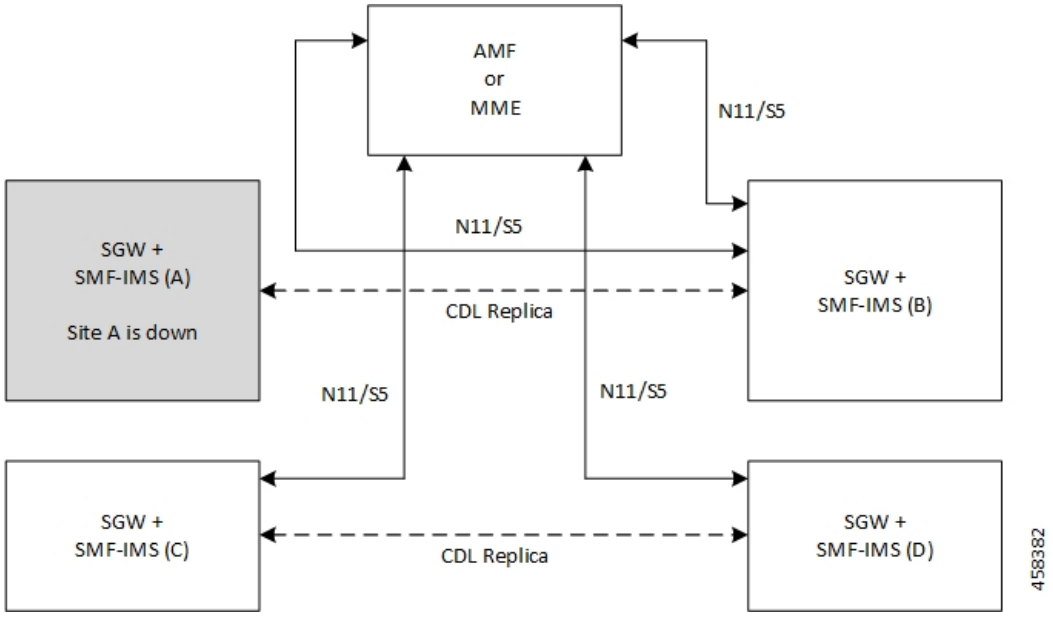
Internal Network Failure

If a functioning K8s cluster has an internal network failure due to a disrupted server communication with the master node, BFD failure, or a K8s pod networking issue, Geo HA is triggered due to K8s dependency checks that are based on the K8s liveliness failure.

In the example shown in the following figure, the AMF or MME transparently starts using the alternate rack server. The N11/S11/S5 and N4/Sxa service addresses are migrated to site B rack B. The system continues signalling from rack B for rack A. At rack B, the session continues without any impact to existing subscriber sessions.

Note Few in-transit calls might fail depending on the state where it is terminated before the UE re-attaches.

Figure 39: Geo HA for Internal Network Failure



Local Switchover

The achieves geo switchover by transparently migrating service IP address to mated peer K8s cluster or rack collocated within the same data center. During the NF start-up, all the K8s cluster Namespaces register with the next-hop BGP router to advertise its service IP address and local IP address along with the priority and route modifier values. Each logical NF exposes separate NF instance toward NRF or DNS, separate configuration, and separate LCM for a Namespace.

Recovery and Failback

For a seamless failover and failback, the UE sessions and the corresponding service IP addresses are grouped together.

The following scenarios describe the seamless failover and failback mechanism for the UE sessions:

- **Normal** - The UE sessions set is created, updated, or deleted from first rack and replicated to second rack.
- **Failure** - The UE sessions set is created, updated, or deleted from second rack and is not replicated to first rack due to its unavailability.
- **Recovery** - The CDL for first rack performs an auto-sync with the CDL for second rack to recover all the UE session data. During the recovery, the second rack continues to handle traffic from the sessions set.

Call Flows

This section describes the key call flows for Dyanmic Routing by Using BGP.

Publish Route for Incoming Traffic in an Active-Standby Mode

The following sections describe the Control Plane and Data Plane call flows in an active/standby mode.

Control Plane Call Flow

This section describes the Control Plane call flow.

Figure 40: Control Plane Call Flow

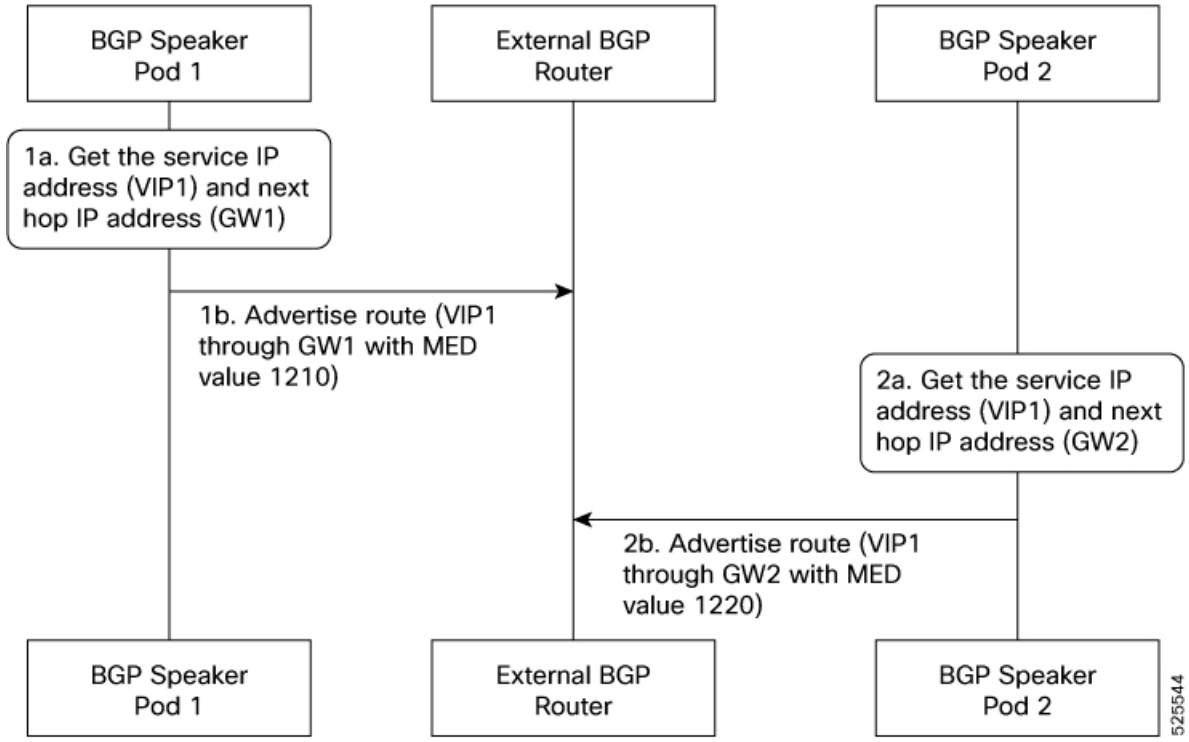


Table 66: Control Plane Call Flow Description

Step	Description
1	The BGP speaker pod starts and fetches the service IP address, next-hop IP address (host IP or loopbackEth), and the Instance ID for the BGP speaker pod. The pod service is exposed through host IP or configured loopbackEth. The NF Instance ID is used to find the route priority or preference.
2	The BGP speaker pod advertises routes by fetching vip-ip (service IP addresses) from the Ops Center.

Data Plane Call Flow

This section describes the data plane call flow.

Figure 41: Data Plane Call Flow

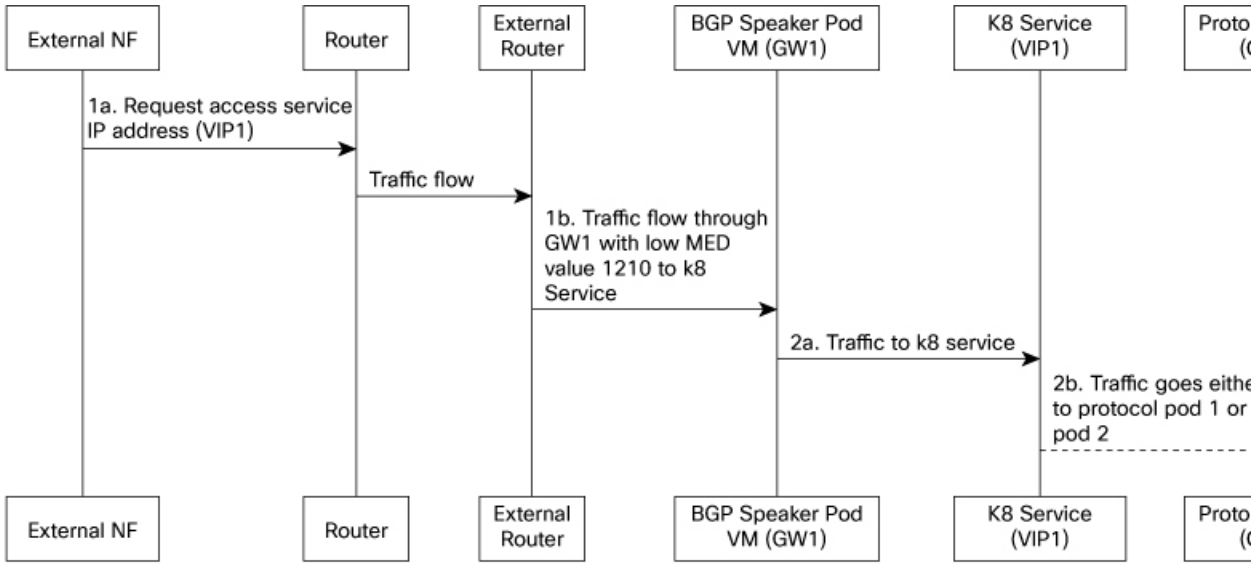


Table 67: Data Plane Call Flow Description

Step	Description
1	External NF requests for service IP address. The request is sent to the nearest connected router through multiple external routers. Then, the router sends the request to the BGP speaker pod with highest priority.
2	The BGP router sets the data plane flow based on the preference value. In the preceding call flow example, the router routes the service request through the host IP1 to pod 1 due to its higher preference value. From host IP1, traffic is forwarded to either sctp pod 1 or pod 2.

Single Protocol Pod Failure Call Flow

The following section describes the Single Protocol Pod Failure call flow.

Figure 42: Single Protocol Pod Failure Call Flow

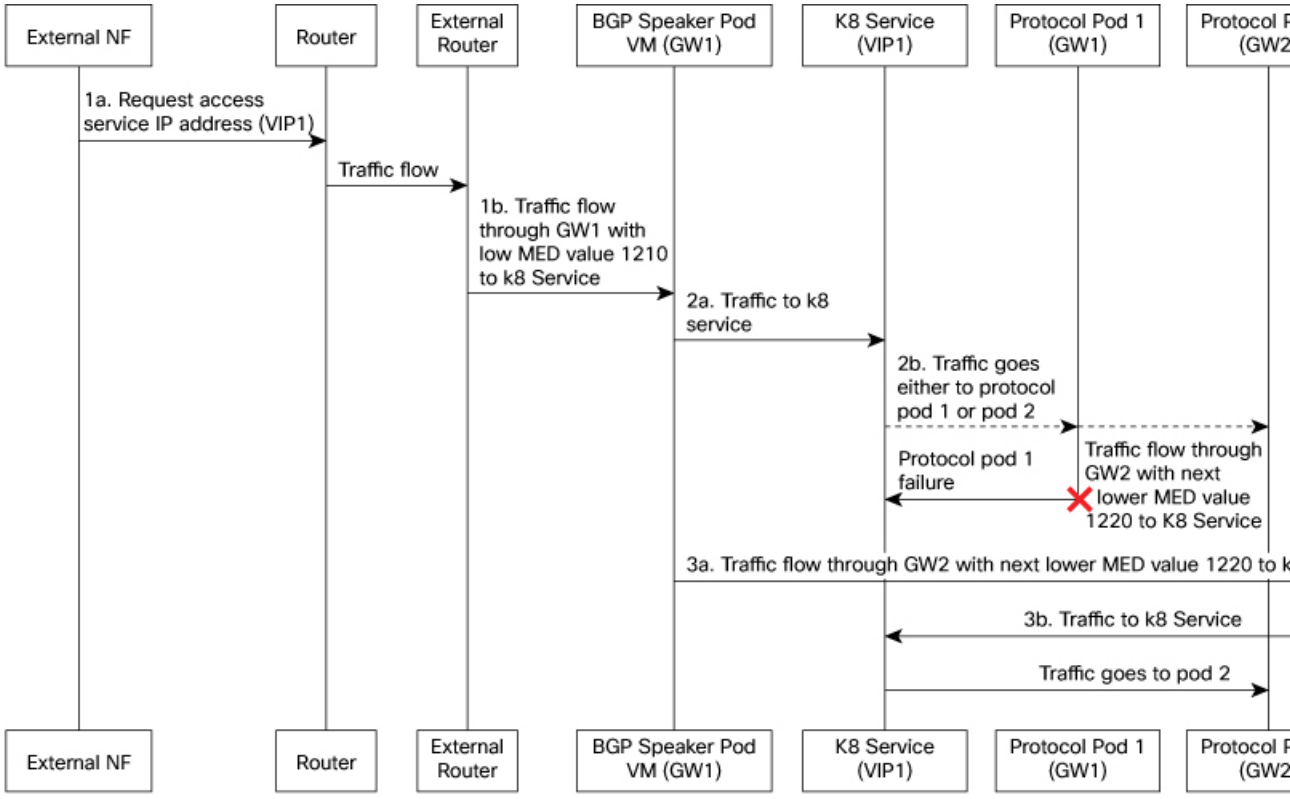


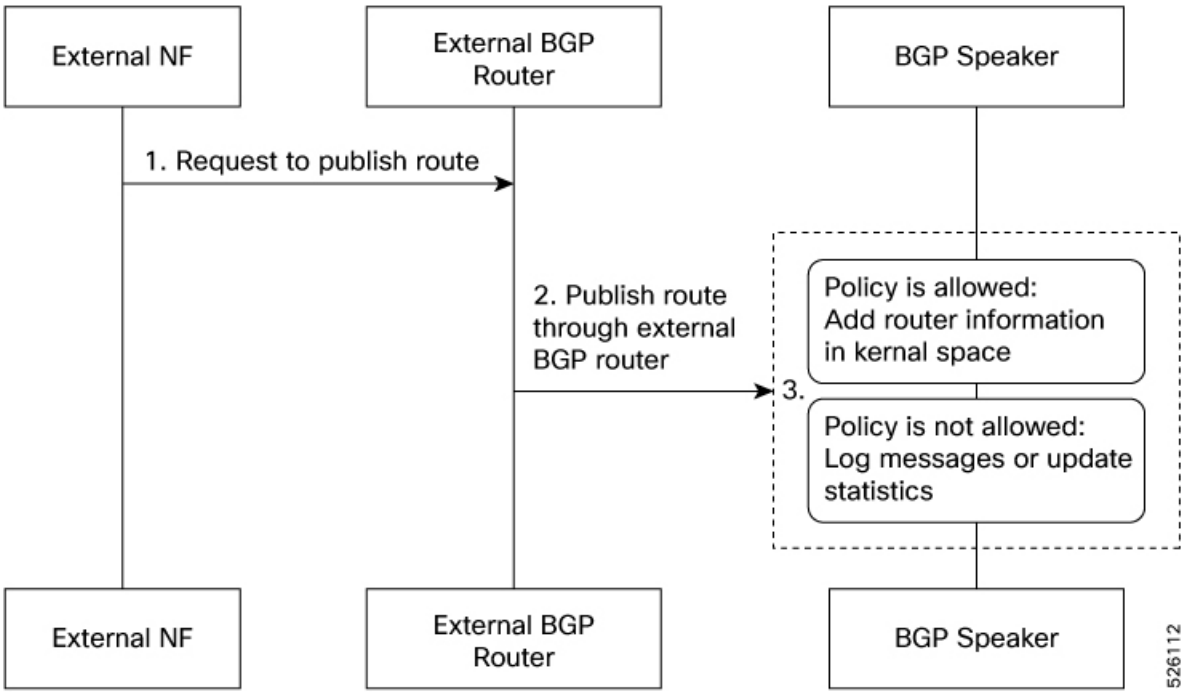
Table 68: Single Protocol Pod Failure Call Flow Description

Step	Description
1	External NF requests for service IP address. The request is sent to the nearest connected BGP router through multiple external routers based on the next highest preference value.
2	The BGP router sets the data plane flow based on the preference value. If the pod with the highest preference value is not available, then the request is routed to the pod with the next highest preference value. In the example shown in the preceding call flow figure, BGP pod 2 with the host IP2 address serves the request due to its higher preference value.

Learn Route for Outgoing Traffic Call Flow

This section describes the learn route for outgoing traffic call flow.

Figure 43: Learn Route for Outgoing Traffic Call Flow



AMF or other systems advertise route to the external BGP route. In turn, the external BGP router advertises routes for its service through BGP.

Table 69: Learn Route for Outgoing Traffic Call Flow Description

Step	Description
1	The BGP speakers receive the routing information.
2	Learn the route by using the BGP protocol.
3	Based on the configure policy, the system either checks the routing information or ignores it.
4	If the policy is not allowed, then the system logs the messages and updates the statistics.
5	The protocol pods configures the route in Kernel space on host through the netlink go APIs.

Configuring Dynamic Routing Using BGP

This section describes how to configure the dynamic routing using BGP.

Configuring AS and BGP Router IP Address

To configure the AS and IP address for the BGP router, use the following commands:

```
config
   router bgp local_as_number
```

```
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.

  In a inter-rack redundancy deployment, you need to configure two Autonomous Systems (AS).

  - One AS for leaf and spine.
  - Second AS for both racks: Rack-1 and Rack-2.

Configure Router ID (Optional)

By default, the BGP speaker uses the IPv4 address of the BGP server running on an interface as the Router ID. BGP speaker uses this IPv4 address as the Router ID for BGP peering for both IPv4 BGP servers and Dual-mode BGP servers.

To configure a custom Router ID, assign an IPv4 address to the loopback (lo) interface on the node where the BGP Speaker pod is running. This assigned IPv4 address is then used as the Router ID.

Note Configuring a Router ID is mandatory if you plan to run the BGP server exclusively with IPv6 addresses (i.e., without any IPv4 address on the interface where the BGP server is running).

Configuring BGP Service Listening IP Address

To configure the BGP service listening IP address, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.

Configuring BGP Neighbors

To configure the BGP neighbors, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
      neighbor neighbor_ip_address remote-as as_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.
- **neighbor** *neighbor_ip_address*—Specify the IP address of the neighbor BGP router.
- **remote-as** *as_number*—Specify the identification number for the AS.

Configuring Bonding Interface

To configure the bonding interface related to the interfaces, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
      bondingInterface interface_name
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.
- **bondingInterface** *interface_name*—Specify the related bonding interface for an interface. If the bonding interface is active, then the BGP gives a higher preference to the interface-service by providing a lower MED value.

Configuring Learn Default Route

If the user configures specific routes on their system and they need to support all routes, then they must set the **learnDefaultRoute** as **true**.

Note This configuration is optional.

To configure the Learn Default Route, use the following commands:

```
config
   router bgp local_as_number
      learnDefaultRoute true/false
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **learnDefaultRoute** *true/false*—Specify the option to enable or disable the **learnDefaultRoute** parameter. When set to true, BGP learns default route and adds it in the kernel space. By default, it is false.

Configuring BGP Port

To configure the Port number for a BGP service, use the following commands:

```
config
   router bgp local_as_number
      loopbackPort port_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **loopbackPort** *port_number*—Specify the port number for the BGP service. The default value is 179.

Policy Addition

The BGP speaker pods learns many route information from its neighbors. However, only a few of them are used for supporting the outgoing traffic. This is required for egress traffic handling only, when cnSGW-C is sending information outside to AMF/PCF. Routes are filtered by configuring import policies on the BGP speakers and is used to send learned routes to the protocol pods.

A sample CLI code for policy addition and the corresponding descriptions for the parameters are shown below.

```
$bgp policy <policy_Name> ip-prefix 209.165.200.225 subnet 16 masklength-range 21..24
as-path-set "^65100"
```

Table 70: Import Policies Parameters

Element	Description	Example	Optional
as-path-set	AS path value	"^65100"	Yes
ip-prefix	Prefix value	"209.165.200.225/16"	Yes
masklength-range	Range of length	"21..24"	Yes
interface	Interface to set as source IP (default is VM IP)	eth0	Yes
gateWay	Change gateway of incoming route	209.165.201.30	Yes
modifySourceIp	Modify source ip of incoming route Default value is False.	true	Yes
isStaticRoute	Flag to add static IP address into kernel route Default value is False.	true	Yes

Monitoring and Troubleshooting

This section describes the show commands that are supported by the Dynamic Routing by Using BGP feature.

show bgp-kernel-route

Use the **show bgp-kernel-route** command to view all the kernel level routes for a BGP router.

The following configuration is a sample output of the **show bgp-kernel-route** command:

```
kernel-route

-----bgpspeaker-pod-1 ----

 DestinationIP       SourceIP        Gateway

 209.165.200.235     209.165.200.239   209.165.200.239

-----bgpspeaker-pod-2 ----

 DestinationIP       SourceIP        Gateway

 209.165.200.235     209.165.200.229   209.165.200.244
```

show bgp-global

Use the **show bgp-global** command to view all BGP global configurations.

The following configuration is a sample output of the **show bgp-global** command:

```
global-details

-----bgpspeaker-pod-1 ----
AS:        65000
Router-ID: 209.165.200.239
Listening Port: 179, Addresses: 209.165.200.239
AS:        65000
Router-ID: 209.165.200.232
Listening Port: 179, Addresses: 209.165.200.232

-----bgpspeaker-pod-2 ----
AS:        65000
Router-ID: 209.165.200.235
Listening Port: 179, Addresses: 209.165.200.235
AS:        65000
Router-ID: 209.165.200.246
Listening Port: 179, Addresses: 209.165.200.246
```

show bgp-neighbors

Use the **show bgp-neighbors** command to view all BGP neighbors for a BGP router.

The following configuration is a sample output of the **show bgp-neighbors** command:

```
neighbor-details

-----bgpspeaker-pod-2 ----
Peer             AS  Up/Down State       |#Received  Accepted
209.165.200.244 60000 00:34:20 Establ      |       10        10
Peer           AS  Up/Down State       |#Received  Accepted
209.165.200.250 60000 00:34:16 Establ      |        3         3

-----bgpspeaker-pod-1 ----
Peer             AS  Up/Down State       |#Received  Accepted
209.165.200.244 60000 00:33:53 Establ      |       10        10
Peer           AS  Up/Down State       |#Received  Accepted
209.165.200.250 60000 00:33:53 Establ      |        3         3
```

show bgp-neighbors *ip*

Use the **show bgp-neighbors** *ip* command to view details of a neighbor for a BGP router.

The following configuration is a sample output of the **show bgp-neighbors** *ip* command:

```
neighbor-details

-----bgpspeaker-pod-1 ----
BGP neighbor is 209.165.200.244, remote AS 60000
  BGP version 4, remote router ID 209.165.200.244
  BGP state = ESTABLISHED, up for 00:34:50
  BGP OutQ = 0, Flops = 0
  Hold time is 90, keepalive interval is 30 seconds
  Configured hold time is 90, keepalive interval is 30 seconds

  Neighbor capabilities:
    multiprotocol:
        ipv4-unicast:   advertised and received
    route-refresh:      advertised and received
    extended-nexthop:   advertised
        Local:  nlri: ipv4-unicast, nexthop: ipv6
    4-octet-as: advertised and received
  Message statistics:
                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                1          2
    Keepalives:            70         70
    Route Refresh:          0          0
    Discarded:              0          0
    Total:                 72         73
  Route statistics:
    Advertised:             0
    Received:              10
    Accepted:              10

-----bgpspeaker-pod-2 ----
BGP neighbor is 209.165.200.244, remote AS 60000
  BGP version 4, remote router ID 209.165.200.244
  BGP state = ESTABLISHED, up for 00:35:17
  BGP OutQ = 0, Flops = 0
  Hold time is 90, keepalive interval is 30 seconds
  Configured hold time is 90, keepalive interval is 30 seconds

  Neighbor capabilities:
    multiprotocol:
        ipv4-unicast:   advertised and received
    route-refresh:      advertised and received
    extended-nexthop:   advertised
        Local:  nlri: ipv4-unicast, nexthop: ipv6
    4-octet-as: advertised and received
  Message statistics:
                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                1          2
    Keepalives:            71         71
    Route Refresh:          0          0
    Discarded:              0          0
    Total:                 73         74
  Route statistics:
    Advertised:             0
    Received:              10
    Accepted:              10
```

show bgp-route-summary

Use the **show bgp-route-summary** command to view all the route details of a BGP router.

The following configuration is a sample output of the **show bgp-route-summary** command:

```
route-details

-----bgpspeaker-pod-1 ----
Table afi:AFI_IP safi:SAFI_UNICAST
Destination: 5, Path: 5

-----bgpspeaker-pod-2 ----
Table afi:AFI_IP safi:SAFI_UNICAST
Destination: 5, Path: 5
```

show bgp-routes

Use the **show bgp-routes** command to view all the routes for a BGP router.

The following configuration is a sample output of the **show bgp-routes** command:

```
bgp-route

-----bgpspeaker-pod-1 ----
   Network              Next Hop             AS_PATH       Age        Attrs
*> 209.165.200.235/24   209.165.200.250      60000         00:36:39   [{Origin: i} {Med:
0}]
*> 209.165.200.227/32   209.165.200.232                 00:36:44   [{Origin: e} {LocalPref:
 220} {Med: 3220}]
*> 209.165.200.247/24   209.165.200.250      60000         00:36:39   [{Origin: i} {Med:
0}]
*> 209.165.200.251/24   209.165.200.250      60000         00:36:39   [{Origin: i} {Med:
0}]
*> 209.165.200.252/32   209.165.200.232                 00:36:44   [{Origin: e} {LocalPref:
 220} {Med: 3220}]

-----bgpspeaker-pod-2 ----
   Network              Next Hop             AS_PATH        Age        Attrs
*> 209.165.200.235/24   209.165.200.250      60000          00:37:02   [{Origin: i} {Med:
0}]
*> 209.165.200.227/32   209.165.200.246                     00:37:11   [{Origin: e}
{LocalPref: 220} {Med: 3220}]
*> 209.165.200.228/24   209.165.200.234      60000          00:37:02   [{Origin: i} {Med:
0}]
*> 209.165.200.229/24   209.165.200.234      60000          00:37:02   [{Origin: i} {Med:
0}]
*> 209.165.200.230/32   209.165.200.246                     00:37:11   [{Origin: e}
{LocalPref: 220} {Med: 3220}]
```

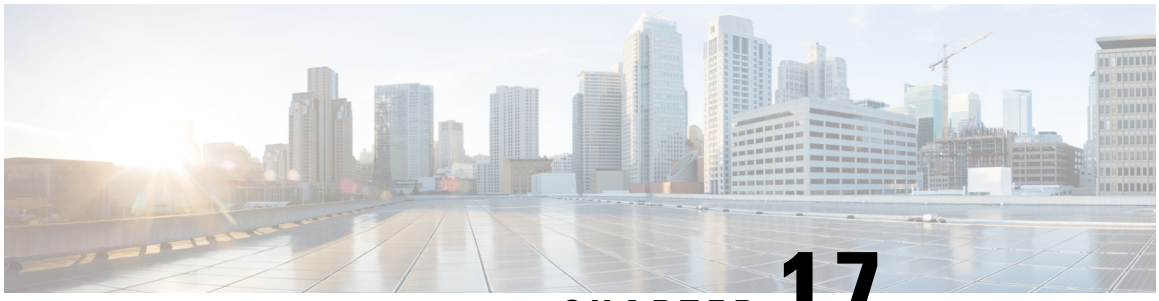
KPIs

The following KPIs are supported for this feature:

Table 71: Statistics for Dynamic Routing by Using BGP

KPI Name	Type	Description/Formula	Label
bgp_outgoing_route request_total	Counter	Total number of outgoing routes.	local_pref, med, next_hope, service_IP

KPI Name	Type	Description/Formula	Label
bgp_outgoing_failedroute request_total	Counter	Total number of failed outgoing routes.	local_pref, med, next_hope, service_IP
bgp_incoming_route request_total	Counter	Total number of incoming routes.	interface, next_hope, service_IP
bgp_incoming_failedroute request_total	Counter	Total number of failed incoming routes.	interface, next_hope, service_IP
bgp_peers_total	Counter	Total number of peers added.	peer_ip, as_path
bgp_failed_peerstotal	Counter	Total number of failed peers.	peer_ip, as_path, error



CHAPTER 17

Emergency Call Support

- Feature Summary and Revision History, on page 185
- Feature Description, on page 185
- How it Works, on page 186
- OAM Support, on page 188

Feature Summary and Revision History

Summary Data

Table 72: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 73: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

Emergency services refer to the functionalities provided by the serving network when the network is configured to support emergency services. These are provided to support IMS emergency sessions.

The MME Emergency Configuration Data contains the Emergency APN which is used for deriving a PDN GW. The MME Emergency Configuration Data can also contain the statically configured PDN GW for the Emergency APN.

cnSGW-C considers calls as emergency when:

- Create Session request has IMEI only.
- The Indication flag indicates unauthenticated IMSI and there's a valid IMSI and IMEI in the Create Session Request.



Note With an emergency session setup, cnSGW-C rejects any additional PDN request (Create Session Request) sent by the MME.

Limitations

This feature has the following limitations in 2021.02.0 and later releases:

- IMEI with 15 digits or 16 digits is supported only for the following procedures—show subscriber, clear subscriber, and monitor subscriber.

How it Works

This section describes how this feature works.

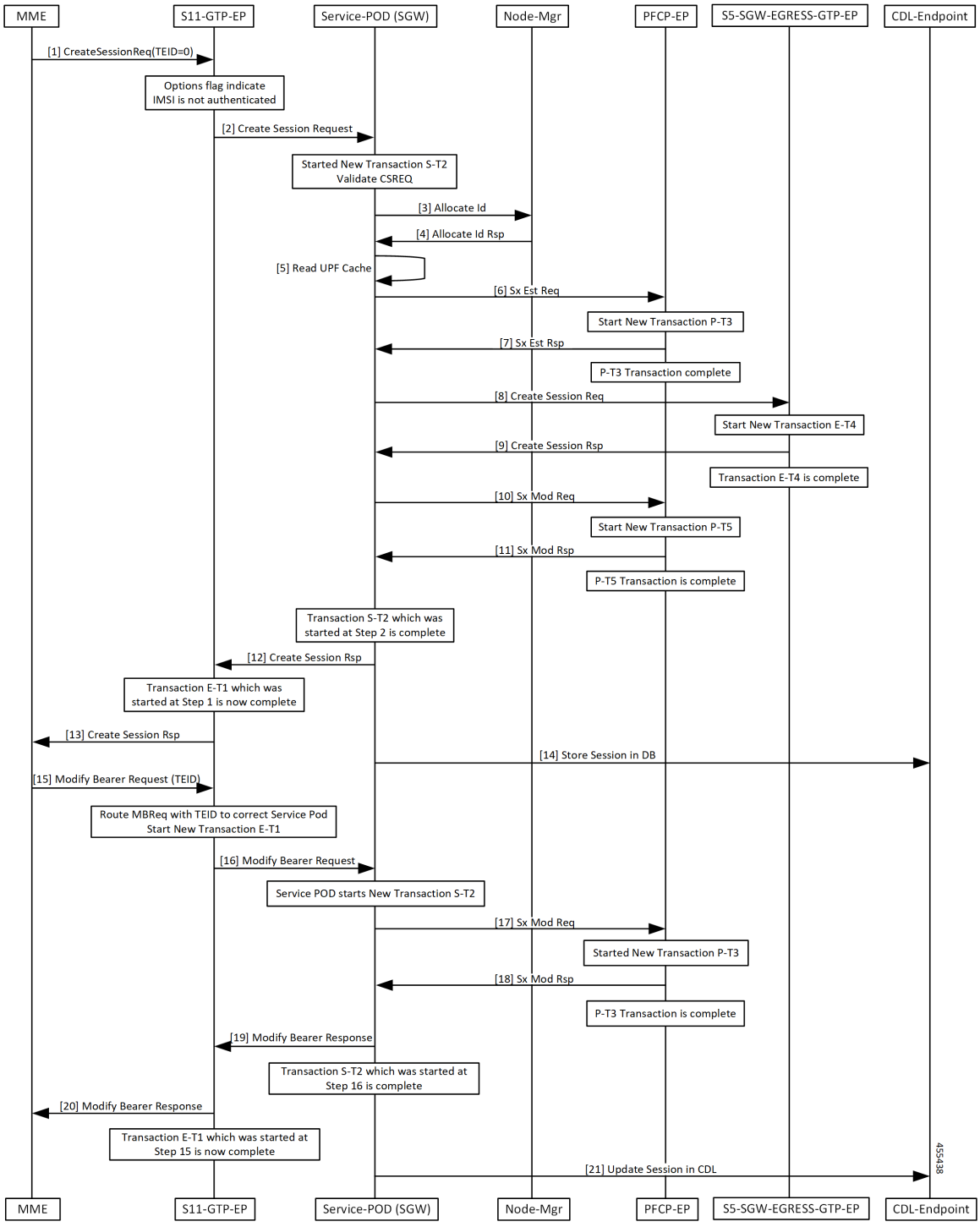
Call Flows

This section describes the key call flows for the feature.

Create Emergency Session Call Flow

This section describes the create emergency session (IMEI, Unauthenticated IMSI) call flow.

Figure 44: Create Emergency Session Call Flow



For an Emergency call, cnSGW-C receives an Initial Attach Request in CSR with an UnAuthenticated IMSI, or with an IMEI only. cnSGW-C allocates TEID and SEID from the Node Manager and sends the Sx Establishment Request with local SEID to the UP to establish the session.

Once cnSGW-C receives the Sx Establishment Response from the UP with the UP SEID and the local GTPU TEID for S5 and S1 GTPU FTEID, cnSGW-C sends the Create Session Request to the PGW using EGTP EP. After receiving response from the PGW for Create Session Response, cnSGW-C sends the Sx Modification Request to connect S5-GTPU tunnel between PGW-U and SGW-U. On successful reception of Sx Modification Response, cnSGW-C sends the Create Session Response to the MME and the session is created in CDL.

With Initial attach procedure, cnSGW-C supports handling of Modify Bearer Request which connects S1 GTPU tunnel between eNodeB and SGW-U. When cnSGW-C receives MBR, it sends Sx Modification Request to connect S1 GTPU tunnel between eNodeB and SGW-U. After receiving Sx Modification Response, cnSGW-C sends Modify Bearer Response to the MME. Session is updated in CDL as the end of transaction.

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics Support

Emergency Counters

```
sgw_ue_stats{app_name="SMF",cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",rat_type="EUTRAN",service_name="sgw-service",status="emergency_release"}
 9
```

```
sgw_ue_stats{app_name="SMF",cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",rat_type="EUTRAN",service_name="sgw-service",status="emergency_setup"}
 9
```

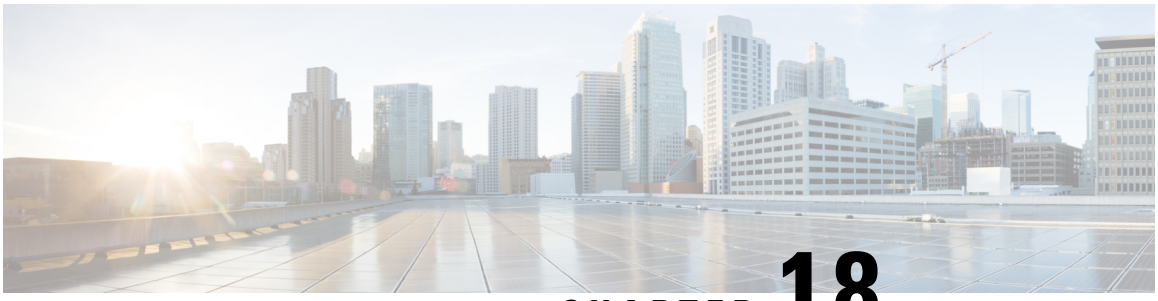
Emergency Statistics

```
db_records_total{app_name="datastore-ep",cluster="session",data_center="test",db=
"session",instance_id="3232257055",service_name="datastore-ep",session_type="SGW:
emergency_call:true",sliceName="1",systemId=""} 1
```

```
db_records_total{app_name="datastore-ep",cluster="session",data_center="test",db=
"session",instance_id="3232257055",service_name="datastore-ep",session_type="SGW:
rat_type:EUTRAN",sliceName="1",systemId=""} 3
```

```
db_records_total{app_name="datastore-ep",cluster="session",data_center="test",db=
"session",instance_id="3232257055",service_name="datastore-ep",session_type="SGW:
state:active",sliceName="1",systemId=""} 3
```

```
db_records_total{app_name="datastore-ep",cluster="session",data_center="test",db=
"session",instance_id="3232257055",service_name="datastore-ep",session_type="total",
sliceName="1",systemId="1"} 3
```



CHAPTER 18

Enhanced PFCP Association Release Procedure for Graceful Session Termination

- Call Disconnection Notification from UPF to cnSGWc through PFCP Association Release, on page 190
- UPF-initiated PFCP Session Release, on page 191
- UPF-initiated Enhanced PFCP Association Release, on page 192
- Enabling EPFAR to Initiate the PFCP Session Release Procedure, on page 192
- Bulk Statistics, on page 193

Call Disconnection Notification from UPF to cnSGWc through PFCP Association Release

Table 74: Feature History

Feature Name	Release	Description
Notification for UPF-initiated PFCP Association Release	2024.03.0	This feature enables cnSGWc to receive notification on UPF-initiated PFCP Association Release Session procedure. This notification indicates to clear the sessions and association simultaneously in UPF and cnSGWc. If the cnSGWc is not notified, the call remains connected until UPF receives the next Session Modify Request from cnSGWc. This leads to loss of subscriber usage reports. Here, the Enhanced PFCP Association Release feature (EPFAR) improves the signalling efficiency and effective handling of usage reports by cnSGWc. **Default Setting:** Disabled – Configuration Required to Enable

When the UPF decides to clear the call due an error or a partial failure, the UPF clears the calls locally without informing the cnSGWc on the call clearance. The call remains connected until the next Session Modify Request is received by the UPF from cnSGWc.

To avoid losing the usage reports and improving the signalling efficiency during call clearance, the Enhanced PFCP Association Release (EPFAR) feature is applied for the UPF to initiate the PFCP Association Release to gracefully clear the session between cnSGWc and UPF simultaneously. This feature complies with the Release 16.9.0 of 3GPP TS 29.244, section 5.18.1 and section 5.18.2.

EPFAR Negotiation

EPFAR feature negotiation processes the communication between the CP function and UP function. When both the CP function and the UP function support the EPFAR feature, the association release action is accomplished. You can also enable the EPFAR feature using the configuration command when the UPF needs to release the association with the SMF.

SGW indicates the support of EPFAR feature by setting the EPFAR bit in CP Function Features IE in the PFCP Association Setup Response towards UPF.

In case of rolling upgrade, whenever the chassis is becoming Active, UPF or cnSGWc sends PFCP Association Update with the UP or CP Function Features, so that EPFAR feature can be negotiated again.

The UPF initiated release is triggered using these procedures:

1. UPF-initiated PFCP Session Release
2. UPF-initiated Enhanced PFCP Association Release

UPF-initiated PFCP Session Release

1. UPF enables the feature for a peer node only if it is negotiated during Association Setup procedure. When the UPF needs to delete a PFCP sesssion due to an error or a partial failure, it initiates the PFCP Session Report requests for the affected session.
2. cnSGWc receives PFCP Session Report Request from UPF with the Report Type and User Report Trigger.
   - The Report Type is set as USAR (Usage Report) when there is a non-zero usage report for the PFCP session or UISR (UP Initiated Session Request) if there is no usage report to send.

     The fifth bit of Octet 6 in Report Type IE is a proprietary IE bit used for indicating UISR for PFCP Session Report Request.
   - User Report Trigger is set as TEBUR (Termination By UP function Report) for a non-zero usage report.
3. UPF sends the Cause IE to the peer node in PFCP Session Report Request message. The IE uses the following values for communicating the Disconnect Reason for session deletion:
   - 201 - Subscriber Clear
   - 202 - Association Release initiated by UP
   - 203 - Recovery Failure
   - 205 - PFCP Cause Self Protection Termination

   The Gz interface carries these causes from cnSGWc:

Cause from UPF	Gz CauseForRecClosing
Subscriber Clear (201)	Normal Release
Association Release initiated by UP (202)	Normal Release
Recovery Failure (203)	Abnormal Release
IP Source Violation (204)	Normal Release
PFCP Cause Self Protection Termination (205)	Normal Release

4. UPF sets the PSDBU (PFCP Session Deleted By the UP function) flag as 1 to indicate the PFCP session deletion.
5. cnSGWc handles these Usage Reports as a part of the release procedure:
   - TEBUR with PSDBU

- Session Reports with UISR and PSDBU

6. The Usage Reports are reported together to CGF in session release.
7. cnSGWc sends the PFCP Session Report Response to UPF. Once the sessions are deleted in locally in UPF, the cnSGWc continues the disconnect procedure.

UPF-initiated Enhanced PFCP Association Release

1. When both the cnSGWc and UPF support EPFAR feature, UPF initiates the PFCP Session Report requests for all the sessions affected by the association release and deletes the sessions.
2. cnSGWc receives the PFCP Association Update Request with PARPS (PFCP Association Release Preparation Start) flag set from the UPF on PFCP Association Release and cnSGWc stops selecting the UPF.
3. cnSGWc receives usage reports with the following flags set from UPF:
   - TEBUR with PSDBU for all sessions with final non-zero usage reports.
   - Session Reports with UISR and PSDBU flags for all sessions with no usage reports.
4. cnSGWc receives PFCP Association Update Request from UPF with the URSS flag set to 1 that indicates all the non-zero Usage Reports for the affected PFCP Sessions are sent. cnSGWc deletes all the remaining Sessions for this UPF.
5. After all the sessions for the UPF are cleared, the cnSGWc triggers PFCP Association Release procedure towards UPF to clear the PFCP Association.

Enabling EPFAR to Initiate the PFCP Session Release Procedure

You can enable the EPFAR feature by using the CLI procedure.

Procedure

Step 1 Enter the **profile converged-core cc supported-features** command in the configuration mode.

Example:

```
[sgw] sgw(config)# profile converged-core cc supported-features [ epfar ]
```

Note epfar—enables support for Enhanced PFCP Association Release

Step 2 Verify the enabled EPFAR feature using the **show running-config profile converged-core** show command.

Example:

```
[sgw] sgw# show running-config profile converged-core
Mon May  6  03:09:36.813 UTC+00:00
profile converged-core cc
 supported-features [ epfar ]
exit
[sgw] sgw#
```

Bulk Statistics

The following statistics are updated with the new labels:

- sgw_pdn_disconnect_stats, sgw_ue_disconnect_stats

  Example Query1:

  ```
  sgw_pdn_disconnect_stats{app_name="smf",
  cluster="Local",data_center="DC",gr_instance_id="1",
  instance_id="0",pdn_type="ipv4",rat_type="EUTRAN",
  reason="userplane_initiated_clear_subscriber",
  service_name="sgw-service"} 1
  ```

  Example Query 2:

  ```
  sgw_ue_disconnect_stats{app_name="smf",cluster="Local",
  data_center="DC",gr_instance_id="1",instance_id="0",
  peer_ip="",reason="userplane_initiated_clear_subscriber",
  service_name="sgw-service"} 1
  ```

  Labels:

  - userplane_initiated_clear_subscriber
  - userplane_initiated_association_release
  - userplane_initiated_recovery_failure
  - sx_urss_association_release
  - userplane_initiated_self_protection_termination

- sgw_service_stats

  Example Query 1:

  ```
  sgw_service_stats{app_name="smf",cluster="Local",
  data_center="DC",fail_reason="",gr_instance_id="1",
  instance_id="0",interface="interface_sgw_egress",
  reject_cause="",service_name="sgw-service",
  sgw_procedure_type="upf_initiated_clear_subscriber",
  status="attempted",sub_fail_reason="",svc_to_svc="False"} 1
  ```

  Example Query 2:

  ```
  sgw_service_stats{app_name="smf",
  cluster="Local",data_center="DC",
  fail_reason="",gr_instance_id="1",
  instance_id="0",interface=
  "interface_sgw_egress",reject_cause="",
  service_name="sgw-service",sgw_procedure_type=
  ```

```
"upf_initiated_clear_subscriber",status="success",
sub_fail_reason="",svc_to_svc="False"} 1
```

Example Query 3:

```
sgw_service_stats{app_name="smf",
cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface=
"interface_sgw_ingress",reject_cause="",
service_name="sgw-service",sgw_procedure_type=
"upf_initiated_clear_subscriber",status="attempted",
sub_fail_reason="",svc_to_svc="False"} 1
```

Example Query 4:

```
sgw_service_stats{app_name="smf",
cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface=
"interface_sgw_ingress",reject_cause="",
service_name="sgw-service",sgw_procedure_type=
"upf_initiated_clear_subscriber",status="success",
sub_fail_reason="",svc_to_svc="False"} 1
```

Labels:

- upf_initiated_clear_subscriber
- upf_initiated_association_release
- upf_recovery_failure
- sx_urss_association_release_initiated_deletion
- userplane_initiated_self_protection_termination

- sgw_sx_session_report_stats

Example Query 1:

```
sgw_sx_session_report_stats{app_name="smf",
cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",reason="psdbu",service_name="sgw-service",
status="success",sx_session_report_type="USAR"} 1
```

Example Query 2:

```
sgw_sx_session_report_stats{app_name="smf",
cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",reason="psdbu",service_name="sgw-service",
status="success"sx_session_report_type="UISR"} 1
```

Labels:

- USAR
- UISR

- sgw_sx_usage_report_stats

Example Query:

```
sgw_sx_usage_report_stats{app_name="smf",
cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",service_name="sgw-service",status="success",
sx_usage_report_trigger_type="TEBUR"} 1
```

Label:

- TEBUR

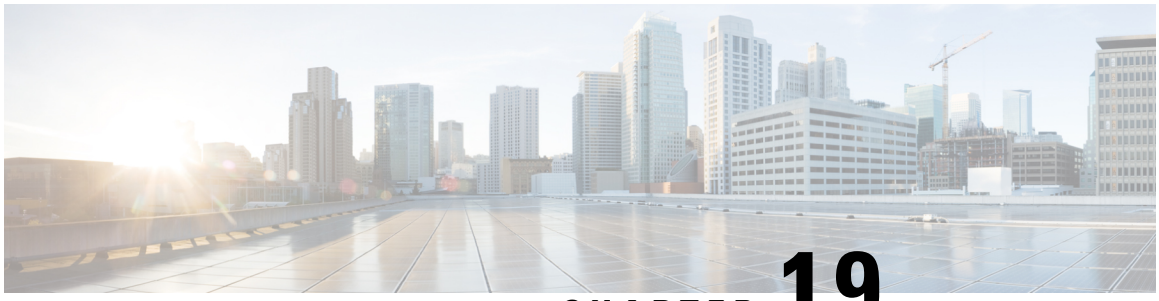
- nodemgr_up_pathfail_reasons

Example Query:

```
nodemgr_up_pathfail_reasons {app_name="SMF",
 cluster="SMF",data_center="DC",gr_instance_id="1",
instance_id="0",service_name="nodemgr",
up_pathfail_reason="up_urss_reason_association_release"} 1
```

Label:

- up_urss_reason_association_release



CHAPTER 19

Extended and Non-Standard QCI Values Support and Validation

- Feature Summary and Revision History, on page 197
- Feature Description, on page 197

Feature Summary and Revision History

Summary Data

Table 75: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	

Revision History

Table 76: Revision History

Revision Details	Release
First introduced.	2021.02.3

Feature Description

This feature supports the following:

- Extended and Non-Standard QCI values as part of CSR | CBR | UBR | MBC call flows

- Extended and Non-Standard QCI values for DSCP marking
- Extended and Non-Standard QCI values for VoLTE marking

Validation for Extended and Non-Standard QCI Values

The following Extended QoS Class Identifier (QCI) and Non-Standard QCI (Operator-defined) values are validated for:

- **CSR | CBR | UBR | MBC Call Flows:** Validation is done for the following QCI values:

  Extended QCI: 65, 66, 69, 70, 80, 82, 83

  Non-Standard (Operator-defined) QCI: 128-154



Note As part of CSR | CBR | UBR | MBC call flows, values other than Standard / Extended / Operator-defined QCI range are also accepted. However, those values aren't supported as part of DSCP marking or VoLTE marking CLIs.

- **DSCP Marking:** Validation is done for the following QCI values:

  Extended QCI: 65, 66, 69, 70, 80, 82, 83

  Non-Standard (Operator-defined) QCI: 128-154

  For more information on DSCP Marking, see *DSCP Marking Support* chapter in the *UCC Serving Gateway Control Plane Function - Configuration and Administration Guide*.

Support and Validation for Extended and Non-Standard QCI Values for VoLTE Marking

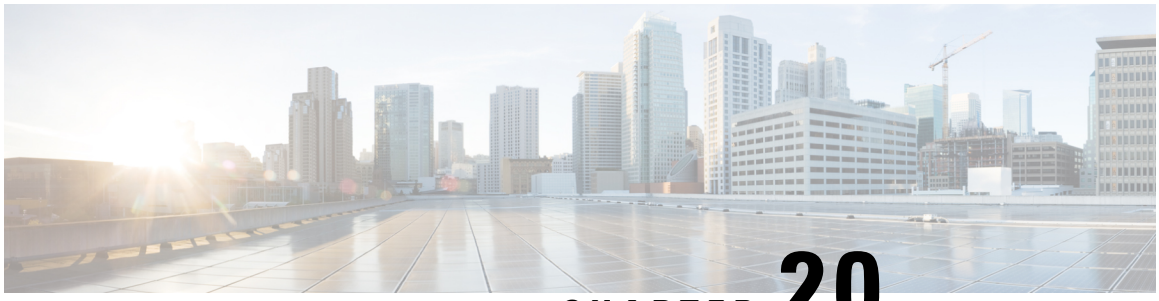
In release prior to 2021.02.2, only Standard QCI values 1-9 were supported for VoLTE Marking.

In 2021.02.2 and later releases, support is added and validated for the following Extended and Non-Standard QCI values:

Extended QCI: 65, 66, 69, 70, 80, 82, 83

Non-Standard (Operator-defined) QCI: 128-154

For more information on VoLTE Marking, see *VoLTE Call Prioritization* chapter in the *UCC Serving Gateway Control Plane Function - Configuration and Administration Guide*.



CHAPTER 20

eMPS and WPS Support

- Feature Summary and Revision History, on page 199
- Feature Description, on page 199
- eMPS/WPS Support, on page 200
- Feature Configuration, on page 205
- OAM Support, on page 209

Feature Summary and Revision History

Summary Data

Table 77: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 78: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

This feature supports the following:

- Enhanced Multimedia Priority Service (eMPS) or Wireless Priority Service (WPS)
- eMPS GTPv2 Load/Overload Self Protection Exclusion
- Message Priority Profiles to define priority either at global level or at each interface level (PFCP, GTP).

eMPS/WPS Support

Feature Description

This feature supports identifying the eMPS subscriber. The feature sets the message priority bit for:

- PFCP interface towards the UP.
- GTPC interface towards the MME and PGW.

This feature includes DSCP marking for request messages in control messages as per the configured value in the profile for eMPS subscriber.

eMPS GTPv2 Load/Overload Self Protection Exclusion Support

Feature Description

cnSGW-C supports interaction of eMPS with GTPv2 load or overload feature. It supports excluding eARPs, APNs, Emergency, or WPS calls during self-protection mode in GTPv2 load or overload feature.

By default, cnSGWc throttles the incoming GTPC messages in the self-protection mode. If cnSGWc is in self-protection mode and the overload exclude profile is configured, then the cnSGWc compares the parameters of the incoming message with the configured DNN, ARP, QCI, and message priority values.

In case of a match or the session is identified as a WPS or an emergency session, cnSGWc processes the incoming message, otherwise rejects it.

cnSGW-C can exclude the dnn-list, arp-list, and qci-list, message-priority from the rejection for incoming request messages in self-protection mode. cnSGW-C excludes this rejection in the following manner:

- Excludes the dnn-list from rejection for any call level procedure when the subscriber APN name (NI+OI) matches with *overload-exclude-profile*.
- Excludes bearer modification or creation from rejection for any new or existing ARP (Priority-Level) value.
- Excludes bearer modification or creation from rejection for any new or existing QCI value.
- Excludes the deleted bearer or the session operations, such as, Delete Bearer Request, Delete Session Request, Delete Bearer command from rejection irrespective of the overload-exclude-profile configuration.

To exclude the incoming GTPC traffic from a specific north-bound interface, see the Configure Overload Exclude Profile section.

Note cnSGW-C does not support message throttling.

Message Priority Profile

Table 79: Feature History

Feature Name	Release Information	Description
Overload Optimization for GTPC Messages using Message Priority Negotiation	2024.04.1	In the overload state, cnSGWc ensures smooth communication by defining the message priority for incoming messages and negotiating the message priority value for outgoing messages. This feature allows the network operator to exempt the incoming high priority messages from throttling by configuring the message priority under the overload exclude profile. This feature also compares the message priority value of the incoming GTPC messages with the configured session message priority value and sets the lowest value in outgoing GTPC messages. **Commands Introduced:** **message-priority { any | s5e | s11 | sxa } upto** *message_priority* **Default Setting:** Disabled—Configuration required to enable

Feature Name	Release Information	Description
Support for Message Priority profiles	2023.04	This feature allows the cnSGWc service to: • Send the Inter-process Communication (IPC) message to the protocol pod for Wireless Priority Service (WPS) session using Priority IPC Stream. • Create the message-priority profile to define priorities either at the global level or at each interface level (PFCP, GTP) **Default Setting**: Disabled – Configuration required to enable.

Feature Description

Priorities determine the order in which service requests are dequeued by a server. For example, the priority that client assigns to individual services can range from 0 to 15, where 0 represents the highest priority.

In cnSGW, the cnSGW service sends the Inter-process Communication (IPC) message to the protocol pod for Wireleass Priority Service (WPS) session using Priority IPC Stream.

cnSGW creates the message-priority profile to define priorities either at global level or at each interface level (PFCP, GTP). You can select the Message Priority value based on:

- ARP and QCI received in Bearer context in the Create Session Request (CSR), Create Bearer Response (CBR), and Create Session Response.
- Update Message priority value based on ARP and QCI received in Bearer context in the Update Bearer Request (UBR).

cnSGW supports the following functionalities:

- WPS sessions per Roaming partner.
- Exclusion of WPS sessions from overload throttling.
- Session type conflict resolution at cnSGW.
- WPS sessions handing at UPF over Sxa Interface for collocated subscriber.
- WPS session monitoring.

Operators can monitor the S-GW service statistics for WPS users and users can also monitor number of active WPS sessions.

How Message Priority Profile Selection Works

Message Priority Functions

The Message Priority mechanism functions are:

- MP (gtp, pfcp) is copied to transactions created for handling an incoming message and the same is copied to child transactions. These priority values are also copied into IPC metadata to send it across to destination pods.
- Protocol pods (protocol, gtp) detect the incoming messages as WPS messages based on the message priority available in the incoming message. These message protocol pods use priority IPC streams toward a service pod.

Conditions for Message Priority Profile Selection

The following conditions apply for Message Priority Profiles selection:

- Subscriber-policy must define group of subscriber and associated Operator policy.
- Operator policy must define the associated DNN policy.
- DNN policy must define a list of DNN and the associated DNN profile.
- DNN profile must have an associated QoS profile name and WPS Profile name.
- QoS Profile must have an associated message-priority-profile.
- If the message-priority-profile is configured in WPS Profile and in S-GW QoS Profile for DNN, then the message-priority-profile configured in WPS profile takes priority.
- If any message priority configuration is changed at run time, it is applicable for the new calls. For existing calls, new configuration is applicable when the Bearer is updated or a new Bearer is added.
- The existing Message Priority gets displayed in outbound messages that are selftriggered, for example, DDN.
- If an incoming PDN level GTP message such as CSReq, MBReq, DSReq, CBReq, DBReq, UBReq is received with valid message priority and if **copy-incoming** is set to true for any of the Bearers in that PDN, then the received messages priority gets copied in all the outgoing GTP/PFCP messages that are triggered by that incoming message.
- If an incoming UE level GTP message such as HO DSReq, RAB is received with valid message priority and **copy-incoming** is set to true for any of the bearers in the PDN, then the received message priority will be copied in all outgoing GTP/PFCP messages(for that PDN) triggered by that incoming message.
- If an incoming PFCP message such as Session report is received with valid messages priority and **copy-incoming** is set to true for any of the Sx bearer in that Sx session, then received messages priority gets copied in outgoing response messages.

Message Priority Selection Logic

- Messages priority value is displayed in outgoing messages based on the following logic:
  - **GTP Messages:**

- **For PDN level messages:** If a PDN have multiple bearers which are marked as WPS (having different GTP message priorities), in that case highest messages priority (0 is highest and 15 is lowest) will be copied into outgoing GTP messages.
- **For UE level GTP messages:** If UE have multiple bearers which are marked as WPS (having different GTP message priorities), in that case the highest messages priority(0 is highest and 15 is lowest) will be copied into outgoing GTP messages.

- **PFCP Messages:** If a Sx session has multiple bearers, which are marked as WPS (having different PFCP message priorities), in that case the highest messages priority (0 is highest and 15 is lowest) gets copied into outgoing PFCP messages.

Message Priority Negotiation for Outgoing GTPC Messages

For outgoing GTPC messages, cnSGWc compares the message priority values received in incoming GTPC messages with the message priority value determined based on configuration and chooses the lowest value. This message priority value is assigned to the outgoing GTPC request or response messages.

Session Type Conflict Resolution at cnSGW

cnSGW resolves a session type conflicts by tagging:

- A session as emergency, if the DNN is tagged to emergency or if the session created is an emergency session. For emergency session, cnSGW uses the QCI/ARP received in the CS request to select a Message Priority.
- A session as IMS/VOLTE, when a bearer gets created with the QCI configured as IMS/VOLTE under DNN profile.
- A session as WPS based on QCI/ARP received.
- If a session satisfies multiple condition, session type is tagged based on the following criteria:

  WPS > Emergency > IMS

Handling of WPS Session at UPF over Sxa Interface

CnSGW handles WPS sessions at UPF over the Sxa Interface in the following ways:

- Selects the message priority that is defined for the PFCP interface under the message-priority profile. For example, Priority value 0 is considered as the highest priority and 15 is considered as the lowest priority.
- This priority gets passed to the UPF in the PFCP header.
- UPF uses this priority during session recovery for early recovery of WPS sessions.



Note For collocated subscribers UPF expects that cnSGW and SMF selects the same message priority for recovering these sessions at the same time.

- If there are multipdn calls, where one of the PDN is WPS and the other PDN is non-WPS, then cnSGW selects WPS Priority for WPS PDN only.

- If there are collocated calls, cnSGW selects WPS priority and SMF selects non-WPS priority for a session. In such cases UPF uses the highest priority for collocated session recovery. This way UPF makes sure that both sessions are recovered at the same time.

WPS Session Monitoring

Use the S-GW Service Statistcs to monitor if the session is WPS. You can tag a session using the session-type values such as “emergency”, “IMS/VOLTE” or “wps”. When WPS bearer is deleted, the WPS tagging is removed and session changes from normal to wps and vice-versa while its active. A new non unique key is added to the session when the session is converted to WPS and the same is removed when it converted back to non-WPS. This is used for **show subscriber wps** and **clear subscriber wps**. cnSGWc supports the **show subscriber count wps**. For more information, refer to the *Monitoring and Troubleshooting* section.

Feature Configuration

Configuring this feature involves the following steps:

- Configure WPS-Profile. For more information, refer to Configuring WPS Profile, on page 205.
- Configuring Message Priority Profiles. For more information, refer to Configuring Message Priority Profiles, on page 206
- Configure SGW-Profile, and enable WPS-Profile and SGW-Profile association. For more information, refer to Configuring WPS-Profile and SGW-Profile Association, on page 207.
- Configure S-GW QoS profiles. For more information, refet to Configuring SGW QoS Profile, on page 208
- Configure DNN-Profile, and enable WPS-Profile and DNN-Profile association. For more information, refer to Configuring WPS-Profile and DNN-Profile Association, on page 207.
- Associate SGW QoS profile with SGW-profile and DNN-Profile. For more information, refer to Associating sgw-qos-profile with sgw-profile and DNN profile, on page 209

Configuring WPS Profile

To configure this feature, use the following configuration:

```
config
  profile wps wps_name
    arp arp_value message-priority-profile msg_priority_profile_name
    dscp dscp_value
    message-priority [ pfcp | gtpc ]
    end
```

NOTES:

- **wps** *wps_name*—Specify the WPS service name. Must be a string.

- **arp** *arp_value*—Specify the range of ARP levels (separated by , or -). Must be an integer or a string. WPS session is decided based on ARP.
- **message-priority-profile** *msg_priority_profile_name*— Specifies that a message-priority profile is added in ARP list within WPS profile. WPS session is decided based on the configured ARP and the associated message priority profile inside the WPS profile.
- **dscp** *dscp_value*—Specify the DSCP marking value in the decimal range 0-63 or hex range 0x0-0x3F. Must be a string.

Configuration Example

The following is an example configuration.

```
config
   profile wps wp1
     arp 2 message-priority-profile message_priority_name
   end
```

Configuration Verification

To verify the configuration:

```
show full-configuration profile wps wps1
profile wps wps1
arp 2 message-priority-profile mp1
exit
```

Configuring Message Priority Profiles

To configure the message priority profile, use the following configuraton.

```
config
  profile  message-priority msg_priority_profile_name
     interface [ any | gtp | pfcp | sbi ] priority [ value range_value |
copy-incoming ]priority copy_incoming_value
  end
```

NOTES:

- **profile message-priority-profile***msg_priority_profile_name*—Specify a message priority profile and a profile name.
- **interface [any | gtp | pfcp | sbi]** —Specify one of the following interface:
  - any
  - gtp
  - pfcp
  - sbi
- **Interface gtp priority [value** *range_value* —Specify the priority value.
  - The Range values from 0 to 31 is for sbi interface and 0 to 15 is for other interfaces.

- If Priority values ranges are from 0 to 15, then 0 indicates the highest priority, while 15 indicates the lowest priority.
- Priority value can be configured per interface. Interface type is also optional and if not configured, same value is applied across all interfaces.

- **Interface gtp priority [value** *range_value* **| copy-incoming]***priority_copy_incoming_value* — If copy-incoming is configured, cnSGW copies the priority received in incoming message to all outgoing messages triggered by that inbound message.

Configuration Verification

To verify the configuration:

```
show full-configuration profile message-priority mp1
profile message-priority mp1
interface gtp priority value 2
interface gtp priority copy-incoming
exit
```

Configuring WPS-Profile and SGW-Profile Association

To configure WPS-Profile and SGW-Profile association, use the following configuration:

```
config
 profile sgw sgw_name
  wps-profile wps_name
  end
```

NOTES:

- **wps-profile** *wps_name*—Specify the Wireless Priority Service (WPS) name. Must be a string.

Configuration Example

The following is an example configuration.

```
config
   profile sgw sgw1
     wps-profile wp1
     end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw
profile sgw sgw1
wps-profile wp1
```

Configuring WPS-Profile and DNN-Profile Association

This section describes how to configure WPS-Profile and DNN-Profile association.

Note If WPS profile is associated with SGW profile and DNN profile, DNN profile takes the priority.

To configure WPS-Profile and DNN-Profile association, use the following configuration:

```
config
 profile dnn dnn_name
   wps-profile wps_name
   end
```

Configuration Example

The following is an example configuration.

```
config
  profile dnn dnn1
  wps-profile wps1
  end
```

Configuration Verification

To verify the configuration:

```
show running-config profile dnn
profile dnn dnn1
wps-profile wps1
```

Configuring SGW QoS Profile

You can associate a message priority profile to SGW QoS profile and to{qci,arp}in the SGW QoS profile. Message priority profile that is associated to {qci,arp} and the one associated to qos profile, and message priority profile associated to WPS are negotiated to find the one which has the better priority.

```
config
  profile sgw-qos-profile  qos_profile_name
    message-priority-profile message_priority_profile_name
    qos qci nonstandard_value arp-priority-level arp_value [
message-priority-profile msg_priority_profile ]
 exit
```

NOTES:

- **message-priorty-profile** *message_priority_profile_name*: Configures message priority profile name
- **qos qci** *standard_value* **arp-priority-level** *arp_value* : Configures QoS parameters for QCI/ARP values. Must be an integer in the range of 1-254. For example, the range values can be 1-9,65,66,69,70,80,82,83,128-254.
- **arp-priority-level** *arp_value* **[message-priority-profile** *msg_priority_profile* **]**: Configures the ARP Priority Level from 1 to 15.

Configuration Example

The following is an example configuration.

```
config
    profile sgw-qos-profile sgw1
      message-priority-profile mp1
      qos qci 2 arp-priority-level 2 message-priority-profile mp1
```

Verification Configuration

To verify the configuration.

```
smf(config)# show full-configuration profile sgw-qos-profile sgw1
profile sgw-qos-profile sgw1
message-priority-profile mp1
qos qci 2 arp-priority-level 2 message-priority-profile mp1
exit
```

Associating sgw-qos-profile with sgw-profile and DNN profile

Use the following sample configuration to associate the WPS profile with the configured DNN profile.

```
config
  profile dnn profile_dnn_name
    wps-profile wps_profile_name
    qci-qos-profile    sgw_qos
    end
```

NOTES:

- **wps-profile** *wps_profile_name*: Enables the wps profile configuration. This profile is configured under the existing DNN profile configuration.
- **qci-qos-profile** *qos_profile_name*: Specify the QoS profile configuration name for S-GW.

Use the following sample configuration to associate the sgw-qos-profile with the configured sgw profile.

```
config
  profile sgw sgw_name
  qci-qos-profile    sgw_qos
  end
```

NOTES:

- **qci-qos-profile** *qos_profile_name*: Specify the QoS profile configuration name for S-GW.

OAM Support

This section describes operations, administration, and maintenance information for this feature

Monitoring and Troubleshooting

This section provides information for WPS session monitoring. Use the following sub-type label in the following show sub CLIs to view details:

show subscriber namespace sgw sub-type *subscriber type*

```
show subscriber nf-service sgw sub-type wps gr-instance 1
Wed Aug  9  09:20:49.500 UTC+00:00
subscriber-details
{
  "subResponses": [
    [
      "id-index:1:0:32768",
      "id-index-key:1:0:globalKey:32768",
      "id-value:16779392",
      "imsi:imsi-123456789012348",
      "msisdn:msisdn-223310101010101",
      "imei:imei-123456786666660",
      "upf:10.1.11.32",
      "upfEpKey:10.1.11.32:10.1.14.119",
      "subscribertype:wps",
      "s5s8Ipv4:10.1.14.19",
      "s11Ipv4:10.1.11.32",
      "namespace:sgw",
      "nf-service:sgw"
    ]
  ]
}
```

show subscriber namespace sgw count sub-type *subscriber type*

You can view count details for the specified subscriber type, SUPI values

```
[sgw] smf# show subscriber namespace sgw count su
Possible completions:
  sub-type
  supi
[sgw] smf# show subscriber nf-service sgw count sub-type wps
Wed Aug  9  09:20:53.760 UTC+00:00
subscriber-details
{
  "sessionCount": 1
}
```

clear subscriber namespace sgw sub-type *subscriber type*

```
[sgw] smf# clear subscriber namespace sgw sub-type ?
Description: Specify Subscriber Type [wps|emergency|volte|non-volte]
Possible completions:
  <string>
[sgw] smf# clear subscriber namespace sgw sub-type wps
Wed Jul  19 10:30:07.716 UTC+00:00
result
ClearSubscriber Request submitted
```

show sessions summary slice-name *slice_name*

```
[sgw] smf# cdl show sessions summary slice-name 1
Wed Aug  9  09:21:08.552 UTC+00:00
session {
    primary-key 2#/#imsi-123456789012348
    unique-keys [ "2#/#16779392" ]
    non-unique-keys [ "2#/#id-index:1:0:32768" "2#/#id-index-key:1:0:globalKey:32768"
"2#/#id-value:16779392" "2#/#imsi:imsi-123456789012348" "2#/#msisdn:msisdn-223310101010101"
 "2#/#imei:imei-123456786666660" "2#/#upf:10.1.11.32" "2#/#upfEpKey:10.1.11.32:10.1.14.119"
 "2#/#subscribertype:wps" "2#/#s5s8Ipv4:10.1.14.19" "2#/#s11Ipv4:10.1.11.32"
"2#/#namespace:sgw" ]
    flags [ flag3:a010b20:a010b20,a010b20:a010e13,
byte-flag1:02:13:03:53:00:00:08:16:0A:01:0B:20:11:FF:01:5B:18:21:63:54:09:2A:21:63:54:00:12:D6:87:10:01:21:63:54:00:00:01:89:D9:99:3F:4E
```

```
 session-state-flag:sgw_active ]
    map-id 1
    instance-id 1
    version 1
    create-time 2023-08-09 09:20:46.302786264 +0000 UTC
    last-updated-time 2023-08-09 09:20:46.867525921 +0000 UTC
    purge-on-eval false
    next-eval-time 2023-08-16 09:20:46 +0000 UTC
    session-types [ SGW:rat_type:EUTRAN SGW:colocated:false SGW:pdn_active:1
SGW:bearer_active:1 SGW:subscriber_type:wps SGW:apn:intershat ]
    data-size 950
}
```

Bulk Statistics Support

The following are the examples for eMPS messages:

```
sgw_pdn_emps_counters{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
service_name="sgw-service",status="active"} 1
```

```
sgw_pdn_emps_stats{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
service_name="sgw-service",status="release"} 7
```

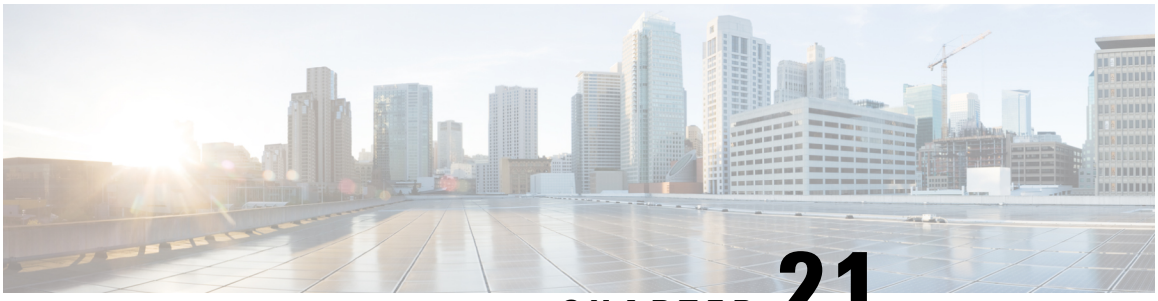
```
sgw_pdn_emps_stats{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
service_name="sgw-service",status="setup"} 8
```

```
gtpc_app_priority_events{app_name="smf",cluster="smf",data_center="smf",event_type=
"NumRxModifyBearerResFrmSerSuccess",instance_id="0",
interface_type="S11",priority_msg="true",service_name="gtpc-ep"} 3
```

```
gtpc_app_priority_events{app_name="smf",cluster="smf",data_center="smf",event_type=
"RxCreateSessionRes",instance_id="0",interface_type="S5E",
priority_msg="true",service_name="gtpc-ep"} 2
```

```
proto_pfcp_msg_total{app_name="smf",cluster="smf",data_center="smf",instance_id="0",
interface_type="SXA",message_direction="outbound",
message_name="N4_MSG_SESSION_ESTABLISHMENT_REQUEST",msgpriority="True",service_name=
"protocol",status="accepted",transport_type="origin"} 2
```

```
proto_pfcp_msg_total{app_name="smf",cluster="smf",data_center="smf",instance_id="0",
interface_type="SXA",message_direction="outbound",
message_name="N4_MSG_SESSION_MODIFICATION_REQUEST",msgpriority="True",service_name="protocol",
status="accepted",transport_type="origin"} 6
```



CHAPTER 21

Failure and Error Handling Support

- Feature Summary and Revision History, on page 213
- Overview, on page 214
- Attach and Detach Failure and Error Handling, on page 214
- Create-Update-Delete Bearer Request and Response Failure and Error Handling, on page 217
- Radio Access Bearer/Modify Bearer Request Failure and Error Handling, on page 222
- Support for Failure Cause Code, Cause Source, and Bearer Context Error, on page 224

Feature Summary and Revision History

Summary Data

Table 80: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 81: Revision History

Revision Details	Release
Added support for the Information Elements (IE): • Failure Cause Code • Cause Source (Bit 1 – CS) • Bearer Context IE error (Bit 2 – BCE)	2021.02.3

Revision Details	Release
First introduced.	2021.01.0

Overview

cnSGW-C supports:

- Attach and Detach failure and error handling.
- Create, Update, Delete Bearer Request and Response failure and error handling.
- Radio Access Bearer or Modify Beare Request failure and error handling.

The different types of failures that can occur during the call processing are as follows, except for Session Setup timer:

- Advance validation failure on request and response.
- Retransmission timeout.
- Transaction service level agreement (SLA).
- Failure reported from peer (UP, PGW, or MME depending on the stage of message process).

For Session Setup timer during attach procedure, following failures can happen:

- Ongoing PDN establishment and Modify Bearer Request from MME isn't received for Initial Attach and multi-PDN.

Attach and Detach Failure and Error Handling

cnSGW-C supports the following:

- Setup timeout functionlaity
- Failure response handling for:
  - Clear Session Request as a part of the Initial Attach and additional PDN setup procedures
  - Delete Beare Request and Delete Session Request processing for the PGW and UPF

Create Session Request Failure Handling

This section covers the Create Session Request procedure failure scenarios.

When failure occurs during Initial Attach procedure, subscriber context isn't created,

When another PDN setup fails, PDN isn't created in subscriber context.

The following table summarizes cnSGW-C behavior during different stages in the call processing for various failure types:

Table 82: cnSGW-C Behavior for Create Session Request Proceudure Failure Scenarios

Scenarios	Sx Signaling (Revert)	S11 Signaling (Revert)	S5 Signaling
• Create Session Request advance validation failure • Sx Session Establishment Response failure from User Plane (UP)	No	Negative - Create Session Response	No
• Create Session Response failure	Yes Delete traffic endpoint for newly created bearers	Negative - Create Session Response	No
• Sx Modify Response process failure	Yes Delete traffic endpoint for newly created bearers	Negative - Create Session Response	Delete Session Request to delete newly created session

Delete Default Bearer Procedure Failure Handling

This section covers the PGW-initiated default bearer deletion procedure failure scenarios.

The following table summarizes cnSGW-C behavior during different stages of call processing for various failure types:

Table 83: cnSGW-C Behavior for Delete Default Bearer Procedure Failure Scenarios

Scenarios	SGW-service Behavior	Signaling	Output
Delete Bearer Request advance validation failure	Send failure/No signaling over Sx/PGW	Negative - S5 Delete Bearer Response	Session/PDN is not deleted
Sx Session Modify failure	Skip failure and continue	Send S11 Delete Bearer Request towards MME S5 Delete Bearer Response (Cause = Accepted) Sx Delete Request (to Delete traffic endpoint) depends upon Sx Session Modify Response	Session/PDN is deleted
S11 Delete Bearer Request failure	Skip failure and continue	Send Delete Bearer Response (Cause = Accepted) to PGW Sx Delete Request (to Delete traffic endpoint)	Session/PDN is deleted

Scenarios	SGW-service Behavior	Signaling	Output
Sx Session Delete Request failure	Skip failure and continue	Send Delete Bearer Response (Cause = Accepted) to PGW	Session/PDN is deleted

Delete Session Procedure Failure Handling

This section covers the MME-initiated Deletion Session procedure failure scenarios.

The following table represents cnSGW-C behavior for the failure scenarios:

Table 84: cnSGW-C Behavior for Delete Session Procedure Failure Scenarios

Scenarios	SGW-service Behavior	Signaling	Output
Delete Session Request advance validation failure	Send failure/No signaling over Sx/MME	Negative - S11 Delete Session Response	Session/PDN is not deleted
Sx Session Modify Request failure	Skip failure and continue	Send Delete Session Request towards PGW Send Delete Session Response (Cause = Accepted) to MME Sx Delete Request (to Delete traffic endpoint) depends upon Sx Session Modify Response	Session/PDN is deleted
S5 Delete Session Request failure	Skip failure and continue	Send Delete Session Response (Cause = Accepted) to MME Sx Delete Request (to Delete traffic endpoint)	Session/PDN is deleted
Sx Session Delete Request failure	Skip failure and continue	Send Delete Session Response (Cause = Accepted) to MME	Session/PDN is deleted

Session Setup Timer during Attach Procedure

This section covers the session setup timer during attach procedure.

The following table represents cnSGW-C behavior for the session timeout scenarios:

Table 85: cnSGW-C Behavior for Session Timeout Failure Scenarios

Scenarios	Sx Signaling (Revert)	S11 Signaling (Revert)	S5 Signaling	Output
Session setup timeout expired after SGW sends Create Session Response to MME and waits for the Modify Bearer Request from MME.	Sx Delete Request (To delete traffic endpoint for newly created bearers)	Delete Session Request	Delete Bearer Request	Session/PDN is deleted

Create-Update-Delete Bearer Request and Response Failure and Error Handling

This section describes create, update, and deletge bearer request and response failure and error handling scenarios.

Create Bearer Procedure Failure Handling

This section covers the PGW-initiated dedicated bearer creation procedure failure scenarios.

The following table summarizes cnSGW-C behavior during different stages of call processing for various failure types:

Note During processing of create dedicated Bearer Request and Response, if SGW receives Context Not Found from peer (MME/UP) it deletes the PDN without performing any signaling towards the peer which sent this cause.

Table 86: cnSGW-C Behavior for Create Bearer Procedure Failure Scenarios

Scenarios	Sx Signaling (Revert)	S11 Signaling (Revert)	S5 Signaling	Output
• Create Bearer Request advance validation failure • Sx Session Modify Request failure (request sent to UP to allocate tunnel endpoint and GTPU TEIDs).	No	No	Negative - Create Bearer Response to PGW	New Bearer Context is not created.

Scenarios	Sx Signaling (Revert)	S11 Signaling (Revert)	S5 Signaling	Output
Create Bearer Request is sent to MME and SGW is waiting for the Response.	Yes To remove traffic endpoint for newly created bearers.	No	Negative - Create Bearer Response to PGW	New Bearer Context is not created. Optional parameters of Create Bearer Response are ignored.
SGW receives Create Bearer Resposne from MME and sends Sx Modify to UP to connect GTPU tunnel between eNodeB and SGW-U.	Yes To remove traffic endpoint for newly created bearers.	Yes To delete newly created bearers.	Negative - Create Bearer Response to PGW	New Bearer Context is not created. Optional parameters of Create Bearer Response are ignored.
Failure in Revert handling for the Delete Bearer Request or Sx Modify Request.	No	No	Negative - Create Bearer Response	Bearer Context is not created. Optional parameters of Create Bearer Response are ignored.

Delete Dedicated Bearer Procedure Failure Handling

This section covers the PGW-initiated dedicated bearer deletion procedure failure scenarios.

The following table summarizes cnSGW-C behavior during different stages of the call processing for various failure types:

Note During processing of delete dedicated Bearer Request and Response if SGW receives Context Not Found from peer (MME/UP), it deletes the PDN without performing any signaling towards the peer which sent this cause.

Table 87: cnSGW-C Behavior for Delete Dedicated Bearer Procedure Failure Scenarios

Scenarios	SGW-Service Behavior	Signaling	Output
Delete Bearer Request advance validation failure	Send failure/No Signaling over Sx/MME	Negative - Delete Bearer Response to PGW	Dedicated Bearer is not deleted.

Scenarios	SGW-Service Behavior	Signaling	Output
Partial Accepted: Delete Bearer Request received with multiple EBI's, where: • Some EBIs belong to PDN • Some EBIs don't belong to PDN/Invalid EBIs	Continue DBR Procedure and delete all existing bearers for which Delete Bearer Request is received. S11 Delete Bearer Request should carry only existing EBIs information Sx Session Modification Request should carry existing EBIs information (Remove Traffic Endpoint)	S5 Delete Bearer Response (Cause = Partially Accepted)	S5 Delete Bearer Response where message level cause is Partially Accepted and bearer level cause is: • **Some EBIs belong to PDN:** S11 Delete Bearer Response • **Some EBIs does not belong to PDN/Invalid EBIs:** Context Not Found CDL is updated (Remove all existing bearers)
Sx Session Modify to set action as DROP	Skip failure and continue	S5 Delete Bearer Response (Cause = Accepted)	Skip failure and continue with DBR Procedure Call Flow: Yes S5 Delete Bearer Response (Cause = Accepted) CDL is updated
S11 Delete Bearer Request Failure	Skip failure and continue	S5 Delete Bearer Response (Cause) = Accepted	Skip failure and continue with DBR Procedure Call Flow: Yes CDL is updated
Sx Session Modify Request failure (Request to remove traffic endpoint on UP)	Skip failure and continue	S5 Delete Bearer Response (Cause = Accepted)	Skip failure and continue with DBR Procedure Call Flow: Yes CDL is updated

Update Bearer Procedure Failure Handling

This section covers the PGW-initiated update bearer procedure failure scenarios.

The following table summarizes cnSGW-C behavior during different stages in call processing for various failure types:

Table 88: cnSGW-C Behavior for Update Bearer Procedure Failure Scenarios

Scenarios	S5/Sx Signaling	Output
Update Bearer Request advance validation failure	Negative - Update Bearer Response	No change in Bearer/PDN context.

Scenarios	S5/Sx Signaling	Output
Update Bearer Request with non-existing EBIs	Update Bearer Response with message level cause as REQ_PARTIALLY_ACCEPTED. Bearer level cause for non-existing EBIs as Context Not Found. (Normal handling for existing EBIs)	Update Bearer Response with message level cause as REQ_PARTIALLY_ACCEPTED. Bearer level cause for non-existing EBIs as CONTEXT_NOT_FOUND. Normal handling for existing EBIs. CDL is updated for existing EBIs only.
Update Bearer Request is sent to MME and waiting for the response	Negative - Update Bearer Response	Negative - Update Bearer Response CDL is not updated.
S11 Update Bearer Response (Message level Cause == CONTEXT_NOT_FOUND) and S5 Update Bearer Req/Rsp had default bearer in the bearer context list	Sx Delete Req/Rsp Negative - Update Bearer Response	Negative - Update Bearer Response CDL is not updated. Statistics/Transactional Logs PDN Key Release + PDN deallocation If this is the last PDN, then resource manager is released, all subscriber keys are released and subscriber deallocation is done.
S11 Update Bearer Response (Message level Cause == CONTEXT_NOT_FOUND) and S5 Update Bearer Req/Rsp didn't have default bearer in the bearer context list	Sx Modify Req/Rsp Negative - Update Bearer Response	CDL is not updated.
S11 Update Bearer Response (Message level Cause == REQ_PARTIALLY_ACCEPTED, Bearer Context Cause == Any failure for dedicated bearer)	Sx Modify Req/Rsp Update Bearer Response with message level cause as REQ_PARTIALLY_ACCEPTED	Update Bearer Response with message level cause as REQ_PARTIALLY_ACCEPTED CDL is updated for successful bearers.

Scenarios	S5/Sx Signaling	Output
S11 Update Bearer Response (Message level Cause == REQ_PARTIALLY_ACCEPTED, Bearer Context Cause == CONTEXT_NOT_FOUND for default bearer)	Sx Delete Req/Rsp Negative - Update Bearer Response	Negative - Update Bearer Response CDL is not updated. PDN Key Release and PDN deallocation If this is the last PDN, then resource manager is released, all subscriber keys are released and subscriber deallocation is done.
If Sx modify is triggered after Update Bearer Response: • Sx Session Modify Request (IPC/Retransmission Timeout/Internal Failure and so on) • Sx Session Modify Response (Cause != ACCEPTED except CONTEXT_NOT_FOUND)	Ignore failure and continue	Ignore failure and continue
If Sx Modify is triggered after Update Bearer Response: • Sx Session Modify Response (Cause == CONTEXT_NOT_FOUND)	[1]	[2]
If Sx Delete is triggered after Update Bearer Response: • Sx Session Delete Request (IPC/Retransmission Timeout/Internal Failure and so on) • Sx Session Delete Response (Cause != ACCEPTED) • Resource manager release (Internal Error)	Ignore failure and continue	Ignore failure and continue

[1] As part of Update Bearer Procedure handling, SGW-service triggers new transaction for PDN deletion:
- Sx Failure Cause received as part of Sx Session Modification Response
  - Context Not Found

SGW Behavior (New Transaction):

- SGW triggers S11 Delete Bearer Request and S5 Delete Session Request to delete that PDN
- No Sx Signaling

SGW Behavior (Update Bearer Transaction): SGW sends S5 Update Bearer Response with Cause as No Resource Available, as part of Update Bearer Procedure Transaction. Also, SGW doesn't initiate any signaling towards UP as soon as it receives Sx Session Modification Response with cause as Context Not Found.

2 As part of Update Bearer Procedure handling, SGW-SVC additionally triggers new transaction for PDN deletion:

- Sx Failure Cause received as part of Sx Session Modification Response
  - Context Not Found

SGW Behavior (New Transaction):

- SGW triggers S11 Delete Bearer Request and S5 Delete Session Request to delete that PDN
- No Sx Signaling

SGW Behavior (Update Bearer Transaction): SGW sends S5 Update Bearer Response with Cause as No Resource Available, as part of Update Bearer Procedure Transaction. Also, SGW doesn't initiate any signaling towards UP as soon as it receives Sx Session Modification Response with cause as Context Not Found.

Radio Access Bearer/Modify Bearer Request Failure and Error Handling

This section covers the Radio Access Bearers (RAB), Modify Bearer Request and Response (MBR) from PGW and User Plane (UP) failure scenarios.

The following table summarizes cnSGW-C behavior during different stages of call processing for various failure types:

Table 89: cnSGW-C Behavior for Radio access Bearer and Modify Bearer Response Procedure Failure Scenarios

Message Type	Failure Interface	Failure Response Recevied	Failure Response to be sent	Handling
MBR initial attach	Sx	CONTEXT_NOT_FOUND	EGTP_CAUSE_ NO_RESOURCES_ AVAILABLE	Cleanup PDN with DSR towards PGW and DBR towards MME
		Other Failure Response	EGTP_CAUSE_ NO_RESOURCES_ AVAILABLE	Cleanup PDN with DSR towards PGW and DBR towards MME. Sx_Modification_Req/ Sx_Session_Delete to cleanup resource on UP
	Timeout	Timeout on PFCP	EGTP_CAUSE_ NO_RESOURCES_ AVAILABLE	Cleanup PDN with DSR towards PGW and DBR towards MME. Sx_Modification_Req/ Sx_Session_Delete to cleanup resource on UP
MBR Service Request	Sx	CONTEXT_NOT_FOUND	EGTP_CAUSE_ NO_RESOURCES_ AVAILABLE	Cleanup PDN with DSR towards PGW and DBR towards MME
		Other Failure Responses	EGTP_CAUSE_ NO_RESOURCES_ AVAILABLE	Do not update anything in PDN, Ignore S5 Signaling
		Timeout	EGTP_CAUSE_ NO_RESOURCES_ AVAILABLE	Do not update anything in PDN, Ignore S5 Signaling
	S5	EGTP_CAUSE_ CONTEXT_NOT_ FOUND	EGTP_CAUSE_ CONTEXT_NOT_ FOUND	Sx_Session_Delete send to UP MBRsp failure to MME (No DBR/DSR)
		Other Failure Responses	Failure Response received from PGW	Do not update PDN/DB
	Timeout	Timeout from PGW	EGTP_CAUSE_ PEER_NOT_ RESPONDING	Do not update PDN/DB

Message Type	Failure Interface	Failure Response Recevied	Failure Response to be sent	Handling
RAB	Sx	CONTEXT_ NOT_FOUND (Single PDN call)	EGTP_CAUSE_ REQ_ACCEPTED	Cleanup PDN with DSR towards PGW and DBR towards MME
		CONTEXT_ NOT_FOUND (Multi PDN call and context not found for one PDN)	Send RAB Resp (EGTP_CAUSE_ REQ_ACCEPTED)	Cleanup the PDN for which context not found received with DSR towards PGW and DBR towards MME Move other PDN/UE to IDLE
		Other Failure Response	EGTP_CAUSE_ REQ_ACCEPTED	Move PDN/UE to IDLE
		Timeout	EGTP_CAUSE_ REQ_ACCEPTED	Move PDN/UE to IDLE

Support for Failure Cause Code, Cause Source, and Bearer Context Error

This section describes the support for the Information Elements (IE) – Failure cause code, Cause Source (CS), and Bearer Context Error (BCE).

For more information on the technical specifications for the IEs, see *3GPP TS 29.274*.

Failure Cause Code

cnSGW-Csupports handling any failure cause code received from SMF on S5 interface.

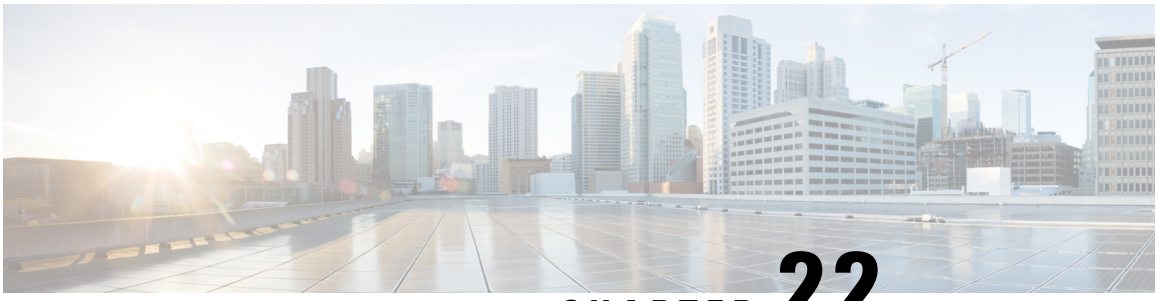
Cause Source

Bit 1 – CS (Cause Source) value ‘0’ indicates that the error is originated by the node sending the message.

Bit 1 – CS (Cause Source) value ‘1’ indicates that the error is originated by the remote node (MME/SGSN to a PGW, or PGW to an MME/SGN).

Bearer Context Error

The default value of Bit 2-BCE (Bearer Context IE Error) is ‘0’. If the BCE Bit is 1, it indicates that the corresponding rejection clause is due to the error in the Bearer Context IE. This bit is discarded if the cause value is one of Acceptance cause value.



CHAPTER 22

GTPC and Sx Path Management

- Feature Summary and Revision History, on page 225
- Feature Description, on page 226
- GTPC and Sx Path Management, on page 226
- Uniformity in compression at N4 and Sx interfaces, on page 231
- GTPC Path Failure, on page 232
- Sx Path Failure, on page 235
- Customization of Path Failure Detection, on page 237

Feature Summary and Revision History

Summary Data

Table 90: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	GTPC and Sx Path Management: Disabled – Configuration required to enable GTPC Path Failure: Enabled – Always-on Sx Path Failure: Enabled – Always-on Path Failure Detection Customization: Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 91: Revision History

Revision Details	Release
Introduced support for IPv6.	2022.04.0
First introduced.	2021.02.0

Feature Description

The GTPC and Sx Path Management feature supports the following:

- GTPC path management using Echo Request and Echo Response messages.
- Sx path management using PFCP Heartbeat Request and Heartbeat Response. Node-level heartbeat procedures between the SGW-C and UPF.
- Detection of the GTPC path failure on S11 and S5 interface.
- Detection of the Sx path failure on the Sx interface.
- Configuration of the path failure detection policy to configure the path failure detection capability.

GTPC and Sx Path Management

Feature Description

GTPC and Sx path sanagement supports the following:

- GTPC peer management using Echo request and Echo response messages exchanged over the S5 and S11 interfaces to monitor peer availability and maintain peer connectivity.
- GTPC Echo request handling from peer network functions and Echo request initiation from the GTPC Pod to verify peer reachability.
- PFCP peer management using association Setup, heartbeat request, and heartbeat response messages over the Sx interface to establish, maintain, and monitor PFCP peer connectivity.
- PFCP session-level signaling to manage session establishment, modification, and deletion between the Service Pod and the user plane.
- ULB-based signaling path management, in which the Unified Load Balancer forwards GTPC and PFCP signaling traffic directly to the appropriate Service Pod for processing while maintaining session affinity throughout the subscriber session lifecycle.

Feature Configuration

Configuring this feature involves the following steps:

- Configure the echo parameters. For more information, refer to Configuring the Echo Parameters, on page 227.
- Configure the heartbeat parameters. For more information, refer to Configuring Heartbeat, on page 227.
- Verify the peer configuration. For more information, refer to Viewing the Peer Configuration, on page 228.

Configuring the Echo Parameters

To configure the Echo parameters, use the following configuration:

Enabling the Echo Request

To enable the Echo Request, use the following configuration:

```
config
  instance instance-id instance_id
    endpoint endpoint_name
      interface [ s11 | s5e ]
        echo interval interval_value
        echo max-retransmissions max_retransmissions_count
        echo retransmission-timeout retransmission_timeout_count
        end
```

NOTES:

- **interval** *interval_value*—Specify the echo interval in seconds. Must be an integer in the range of 60-3600. Default value is 60 seconds.
- **max-retransmissions** *max_retransmissions_count*—Specify the maximum number of retries for GTP Echo Request. Must be an integer in the range of 0-15. Default value is 3.
- **retransmission-timeout** *retransmission_timeout_count*—Specify the Echo Request retransmission timeout period in seconds. Must be an integer in the range of 1-20. Default value is 5.

Disabling the Echo Request

To disable the Echo Request, use the following configuration:

```
config
  instance instance-id instance_id
    endpoint endpoint_name
      replicas replicas_count
      interface interface_name
        no echo
        end
```

Configuring Heartbeat

To configure the heartbeat parameters, use the following configuration:

Enabling Heartbeat

To enable a heartbeat, use the following configuration:

```
config
   instance instance-id instance_id
      endpoint pfcp
         interface sxa
            heartbeat
            interval interval
            retransmission-timeout timeout
            max-retransmissions retransmission_count
            end
```

NOTES:

- **interval** *heartbeat_interval*—Specify the heartbeat interval in seconds. Must be an integer in the range of 0-3600. To disable, set to 0.
- **max-retransmissions** *max_retransmissions*—Specify the maximum number of retries for the PFCP Heartbeat Request. Must be an integer in the range of 0-15. Default value is 4.
- **retransmission-timeout** *retransmission_timeout*—Specify the heartbeat retransmission timeout period in seconds. Must be an integer in the range of 1-20. Default value is 5.

Disabling Heartbeat

To disable a heartbeat, use the following configuration:

```
config
   instance instance-id instance_id
      endpoint pfcp
         interface sxa
            heartbeat
            interval interval
            end
```

NOTES:

- **interval** *heartbeat_interval*—Specify the heartbeat interval as 0 to disable the heartbeat.

Viewing the Peer Configuration

To view the peer restart counter, use the following configuration:

The following command displays the peer configuration:

show peers all [endpoint] [local addr] [peer addr]

```
show peers all SXA 209.165.201.12:8805 209.165.201.18:8805 POD CONNECTED
ENDPOINT LOCAL ADDRESS PEER ADDRESS DIRECTION INSTANCE TYPE TIME RPC ADDITIONAL DETAILS
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
SXA 209.165.201.12:8805 209.165.201.18:8805 Inbound nodemgr-0 Udp 4 hours SGW-U Capacity:
65535,
LoadMetric: 0,LoadSeqNo: 0,Mode: Online,OverloadMetric: 0,OverloadSeqNo: 0,Priority: 65535

show peers all S11 209.165.201.4:2123 209.165.201.7:2123 LOCAL POD CONNECTED
ADDITIONAL ENDPOINT ADDRESS PEER ADDRESS DIRECTION INSTANCE TYPE TIME RPC DETAILS
```

```
--------------------------------------------------------------------------------
-----------------
S11 209.165.201.4:2123 209.165.201.7:2123 Inbound nodemgr-0 Udp 25 seconds MME Recovery:
10

show peers all S5E 209.165.201.4:2123 209.165.201.21:2123 LOCAL POD CONNECTED
ADDITIONAL ENDPOINT ADDRESS PEER ADDRESS DIRECTION INSTANCE TYPE TIME RPC DETAILS
--------------------------------------------------------------------------------
----------------
S5E 209.165.201.4:2123 209.165.201.21:2123 Inbound nodemgr-0 Udp 25 seconds PGW Recovery:
10

show peers all POD CONNECTED
ENDPOINT LOCAL ADDRESS PEER ADDRESS DIRECTION INSTANCE TYPE TIME RPC ADDITIONAL DETAILS
--------------------------------------------------------------------------------
<none> 209.165.201.29 209.165.201.18:8001 Outbound rest-ep-0 Rest 17 hours UDM <none>
<none> 209.165.201.29 209.165.201.18:8002 Outbound rest-ep-0 Rest 17 hours AMF <none>
<none> 209.165.201.29 209.165.201.18:8003 Outbound rest-ep-0 Rest 17 hours PCF <none>
<none> 209.165.201.29 209.165.201.18:8004 Outbound rest-ep-0 Rest 17 hours CHF <none>
<none> 209.165.201.29 209.165.201.18:9040 Outbound rest-ep-0 Rest 17 hours CHF <none>
S11 209.165.201.4:2123 209.165.201.6:2123 Inbound nodemgr-1 Udp 18 minutes MME Recovery:
10
S5E 209.165.201.12:2123 209.165.201.24:2123 Inbound nodemgr-1 Udp 5 hours PGW Recovery:
65535
SXA 209.165.201.12:8805 209.165.201.18:8805 Inbound nodemgr-0 Udp 22 minutes SGW-U Capacity:
 65535,LoadMetric: 0,LoadSeqNo: 0,Mode: Online,OverloadMetric: 0,OverloadSeqNo: 0,Priority:
 65535
```

Configuration Example

The following is an example configuration to enable the echo.

```
config
   instance instance-id 1
      endpoint gtp
         interface s11
            echo interval 60
            echo max-retransmissions 5
            echo retransmission-timeout 4
            end
```

The following is an example configuration to disable the echo.

```
config
   instance instance-id 1
      endpoint gtp
         replicas 1
         interface s5e
            no echo
            exit
         interface s11
            no echo
            end
```

Limitations

Automatic Service Pod and node failure handling is not supported in this release for ULB-enabled deployments.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Alerts

To configure Alerts for Peer Up and Peer Down, see *Key Performance Indicators* chapter in *Cisco Ultra Cloud Serving Gateway Control Plane Function - Metrics Reference*.

Bulk Statistics Support

Node Manager

The following are examples of Echo Transmitted and Echo Retransmitted messages:

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_retx",
gtpc_peer_ip="209.165.201.11",instance_id="1",interface_type="S5E",service_name="nodemgr"}
 3
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_tx",
gtpc_peer_ip="209.165.200.230",instance_id="1",interface_type="S11",service_name="nodemgr"}
 2
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_tx",
gtpc_peer_ip="209.165.201.11",instance_id="1",interface_type="S5E",service_name="nodemgr"}
 4
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_tx_initial",
gtpc_peer_ip="209.165.200.230",instance_id="1",interface_type="S11",service_name="nodemgr"}
 2
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_tx_initial",
gtpc_peer_ip="209.165.201.11",instance_id="1",interface_type="S5E",service_name="nodemgr"}
 1
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_res_rx",
gtpc_peer_ip="209.165.200.230",instance_id="1",interface_type="S11",service_name="nodemgr"}
 2
```

GTPC-EP Pod

The following are examples of Echo Request received and Echo Response sent messages:

```
gtpc_echo_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_rx",
gtpc_peer_ip="209.165.200.230",instance_id="0",service_name="gtpc-ep"} 1
```

```
gtpc_echo_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_rx",
gtpc_peer_ip="209.165.200.231",instance_id="0",service_name="gtpc-ep"} 1
```

```
gtpc_echo_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_req_rx",
gtpc_peer_ip="209.165.201.11",instance_id="0",service_name="gtpc-ep"} 1
```

```
gtpc_echo_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_res_tx",
gtpc_peer_ip="209.165.200.230",instance_id="0",service_name="gtpc-ep"} 1
```

```
gtpc_echo_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_res_tx",
gtpc_peer_ip="209.165.200.231",instance_id="0",service_name="gtpc-ep"} 1
```

```
gtpc_echo_msg_stats{app_name="smf",cluster="cn",data_center="cn",gtpc_msg_type="gtpc_echo_res_tx",
gtpc_peer_ip="209.165.201.11",instance_id="0",service_name="gtpc-ep"} 1
```

Procedure-Level

The following are examples of how to check the incremented values of Heartbeat Request, Heartbeat Response, and Heartbeat Request retry.

```
nodemgr_up_hb_msg_stats{app_name="smf",cluster="Local",current_nodemgr_id="0",data_center="DC",
instance_id="0",interface_type="SXA",primary_nodemgr_id="0",service_name="nodemgr",up_ep_key=
"209.165.201.1:209.165.201.21",up_msg_type="up_heartbeat_req_retx"} 3
```

```
nodemgr_up_hb_msg_stats{app_name="smf",cluster="Local",current_nodemgr_id="0",data_center="DC",
instance_id="0",interface_type="SXA",primary_nodemgr_id="0",service_name="nodemgr",up_ep_key=
"209.165.201.1:209.165.201.21",up_msg_type="up_heartbeat_req_tx"} 5
```

```
nodemgr_up_hb_msg_stats{app_name="smf",cluster="Local",current_nodemgr_id="0",data_center="DC",
instance_id="0",interface_type="SXA",primary_nodemgr_id="0",service_name="nodemgr",up_ep_key=
"209.165.201.1:209.165.201.21",up_msg_type="up_heartbeat_rsp_rx"} 5
```

Uniformity in compression at N4 and Sx interfaces

This feature introduces a peer-level capability exchange mechanism during the PFCP Association Setup procedure. It ensures that compression capabilities are correctly negotiated between the Control Plane (cnSGW-C) and the User Plane Function (UPF), preventing call rejections that occur in converged core deployments when a shared UPF has compression enabled.

Sx protocol compression is enabled by default in CUPS deployments. However, it is not supported on the converged core (cnSGW-C/SMF). When a UPF is shared between a converged core and a CUPS core, call rejections occur because the converged core cannot process compressed PFCP messages. This feature allows the cnSGW-C to explicitly declare its compression support status, ensuring the UPF sends uncompressed messages to converged core peers.

How it works

The feature functions through an exchange of Information Elements (IEs) during the PFCP Association Setup:

1. UPF Capability: The UPF sends the Operator configurable UPF capability IE (Type 359, as defined in 3GPP TS 29.244) in the Association Setup Request.
2. CP Capability: The cnSGW-C (CP) responds with a proprietary Operator configurable CP capability IE (Type 32769, Enterprise ID 9) in the Association Setup Response.

3. Capability Declaration: The cnSGW-C populates this IE with a value of 0 (Ccmpression disabled) to ensure the UPF sends uncompressed messages to this peer.



Note This feature is "Always-on" and auto-negotiated. No manual configuration is required to enable the capability exchange.

Per-Peer capability logic

The UPF determines whether to send compressed or uncompressed messages based on the declaration received from the SMF:

- IE 32769 (Value 0): Sent by SMF/cnSGW → UPF sends uncompressed messages.
- IE 32769 (Value 1): Sent by CUPS-CP → UPF sends compressed messages.

Monitor protocol

To verify the capability exchange, monitor the PFCP Association Setup Response sent by the SMF/cnSGW:

- IE Type: 32769
- Enterprise ID: 9
- Value: 0 (Compression disabled/not supported)

GTPC Path Failure

Feature Description

GTPC path failure detects peer-level GTPC path failure on the S11 and the S5 interface when:

- Echo Response contains a new restart counter value.
- Echo Request contains a new restart counter value.
- Echo Response is not received.
- Create Session Request or Modify Bearer Request contains a new restart counter value.
- Create Session Response or Modify Bearer Response contains a new restart counter value.

The connections may get disconnected due to different path failure is as follows:

- s11_path_failure
- s5e_path-failure
- s11_path_failure_local_purge
- s5e_ path_failure_local_purge

- s5e_recovery
- s11_recovery
- s5e_recovery_local_purge
- s11_recovery_local_purge

How it Works

This section describes how this feature works.

GTPC Path Failure Detection

Path failure is detected in the following conditions:

- Echo Failure: Echo failure occurs when the peer doesn't respond to the Echo Request or the retries.
- Restart Counter in Echo Response or Control Messages: The GTPC entity receives Recovery IE either in an Echo Response or from the peer GTPC message. GTPC entity compares the received the restart counter value with the previously stored restart counter value for that peer entity and performs the following:
  - Stores the received restart counter value for the peer when previously stored value isn't available.
  - When the max-remote-rc-change parameter is not configured, GTPC detects the change in the restart counter.
  - When max-remote-rc-change is configured, calculate the difference in the restart counter value considering restart counter rollover. Detects path failure when the difference between new and old restart counter is less than the value of max-remote-rc-change.

Note For more information on max-remote-rc-change, refer to Customization of Path Failure Detection, on page 237.

Path Failure Handling

Upon detecting a path failure, the network node notifies the failure through the Operation and Maintenance system and performs the following:

- Deletes the PDN connections (EPS bearer contexts) or the associated PDP contexts with peer IP address.
- Specifies the following actions for the selected interface:
  - **Local Purge:** The cnSGW-C clears the affected bearer (or PDN if the default bearer receives the path failure) locally without informing the peer. This action is default for all interfaces.

Note cnSGW-C sends the Sx Session Delete Request to UPF to clear session on path failure detection.

- **Signal-Peer:** The cnSGW-C sends control signal towards the peer MME and P-GW.

  When signaling:

  - For PDN deletion, the SGW sends a Delete Session Request message to the PGW and a Delete Bearer Request (with LBI) message to the MME.
  - SGW sends a Delete Request on the S11 or the S5 interface to notify the peer.



Note Echo Request exchange is stopped when the peer is deleted.

Feature Configuration

Configuring this feature involves the following steps:

- Configure the action that must be taken on path failure detection. For more information, refer to Configuring Action on Path Failure Detection, on page 234.
- Configure the notification to update the peer node. For more information, refer to Configuring Notification to Update the Peer Node, on page 234.
- Verify the configuration. For more information, refer to Configuration Example, on page 234.

Configuring Action on Path Failure Detection

To configure the action for path failure detection, use the following configuration:

```
config
  profile sgw sgw_name
   path-failure [ s11 | s5e ] [ local-purge | signal-peer ]
   end
```

Configuring Notification to Update the Peer Node

Whenever cnSGW-C is restarted, the restart counter needs to be updated. For implementing this functionality, verify the Kubernetes use-volume-claims parameter value is set as true in Ops Center.

This configuration updates the restart counter when cnSGW-C restarts with the CLI system mode shutdown and system mode running.

Configuration Example

The following is an example configuration of path failure detection:

```
config
profile sgw sgw1
 path-failure s11 local-purge
 path-failure s5e local-purge
exit
```

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics Support

The following are examples of the GTPC path failure:

```
nodemgr_gtpc_pathfail_reasons{app_name="smf",cluster="cn",
data_center="cn",instance_id="1",pathfail_reason="pathfail_no_echo_rcv",
service_name="nodemgr"} 2
```

```
nodemgr_gtpc_pathfail_reasons{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pathfail_reason="pathfail_echo_res_rc_change",
service_name="nodemgr"} 1
```

Sx Path Failure

Feature Description

Sx path failure detects the path failure on the Sx interface when:

- Heartbeat Request contains a higher value of recovery timestamp.
- Heartbeat Response contains a higher value of recovery timestamp.
- Heartbeat Response is not received.
- Sx Association Request is received.
- cnSGW-C receives the Sx Association Update Request to release the peer.

How it Works

This section describes how this feature works.

Heartbeat Request

The cnSGW-C or UPF sends the Heartbeat Request on a path to the peer node to find out if the node is alive. The Heartbeat Request messages are sent for each peer with which a PFCP control association is established. cnSGW-C or UPF is prepared to receive the Heartbeat Request and it responds with a Heartbeat Response. The Heartbeat Request starts with the peer when a new session is established with the peer and it’s stopped when the last session is released from the peer.

cnSGW-C and UPF send the Heartbeat Request based on the configured interval. If the peer doesn’t respond, the message is retried for the configured number of times within the retry interval. After the response is received the defined action is taken for the calls associated with the corresponding peer.

Recovery Time Stamp is the IE which contains the start time of the peer node. The Heartbeat Request contains the selfrecovery timestamp value sent to the peer.

Note The heartbeat request is stopped only when the peer is deleted.

Heartbeat Response

The Heartbeat Response message is sent as a response to a received Heartbeat Request.

Recovery Timestamp is the IE which contains the start time of the node. Heartbeat Response contains the peer's Recovery Timestamp value.

Sx Path Failure Detection

Sx path failure is detected in the following conditions:

- **Heartbeat Failure:** When the peer doesn't respond to the heartbeat sent and also to the retries.
- **Recovery Timestamp Change in Heartbeat:** When the Heartbeat Response has a new Recovery Timestamp value then the previously received value. If the Recovery Timestamp value received is lower than the previously received value, the path failure isn't detected.
- **Sx Association Message:** When the Sx Association message is received again from the peer. In this case, all the calls are cleared and a notification is sent to eGTP peer.
- **Sx Association Release Message:** When the Sx Association release message is received. In this case, all the calls are cleared and a notification is sent to eGTP peer.

Path Failure Handling

When the recovery timestamp value received is more than the previously received value, the peer restart is detected. If the timestamp is lower than the previously received value, the value is ignored and peer restart isn't detected.

When the peer restart is detected to indicate the path failure for the peer, all the calls connected to that peer are cleared. The disconnection reason used for such calls is Sx path failure.

Sx association is also removed on detecting Sx path failure.

Heartbeat Handling

Whenever a PFCP entity receives a Heartbeat Request message (even from unknown peers), it responds with a Heartbeat Response message.

After a path failure is detected due to **No response to peer** error, no further Heartbeat Request is sent to that peer until the association is reestablished. Calls are cleared based on the path failure detection policy configuration.

Note After the Sx associations are removed, the heartbeat is stopped when Sx path failure is detected.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following are examples of the procedure-level statistics incremented for Heartbeat Request, Heartbeat Response, and Heartbeat Request retry:

```
nodemgr_up_hb_msg_stats{app_name="smf",cluster="Local",current_nodemgr_id="0",
data_center="DC",instance_id="0",interface_type="SXA",primary_nodemgr_id="0",service_name=
"nodemgr",up_ep_key="209.165.201.5:209.165.201.28",up_msg_type="up_heartbeat_req_retx"}
 3
```

```
nodemgr_up_hb_msg_stats{app_name="smf",cluster="Local",current_nodemgr_id="0",
data_center="DC",instance_id="0",interface_type="SXA",primary_nodemgr_id="0",service_name=
"nodemgr",up_ep_key="209.165.201.5:209.165.201.28",up_msg_type="up_heartbeat_req_tx"}
 5
```

```
nodemgr_up_hb_msg_stats{app_name="smf",cluster="Local",current_nodemgr_id="0",
data_center="DC",instance_id="0",interface_type="SXA",primary_nodemgr_id="0",service_name=
"nodemgr",up_ep_key="209.165.201.5:209.165.201.28",up_msg_type="up_heartbeat_rsp_rx"}
 5
```

Customization of Path Failure Detection

Feature Description

cnSGW-C lets you configure the path failure detection policy. By default, the path failure detection policy is enabled.

- **GTPC Path Failure Detection Customization:** GTPC path failure is detected when:
  - The Echo Request retries are exhausted.
  - The Echo Request or Response Restart counter is modified.
  - The control message Response Restart counter is modified.
  - If the absolute difference between the new and old restart counters is less than the value configured for max-remote-rc-change.

Note GTPC Path Failure Detection Customization allows user to ignore false peer restart with max remote restart counter (max-remote-rc-change) change functionality.

- **Sx Path Failure Detection Customization:** PFCP path failure is detected when:

- The Heartbeat Request retries are exhausted.
- The Heartbeat Request or Response recovery timestamps have modified.

Feature Configuration

Configuring this feature involves the following steps:

- Configure the GTPC path failure customization. For more information, refer to Configuring GTPC Path Failure Customization, on page 238.
- Configure the Sx path failure customization. For more information, refer to Configuring Sx Path Failure Customization, on page 238.

Configuring Sx Path Failure Customization

To configure the Sx path failure customization, use the following configuration:

```
config
  policy sx-path-failure-detection policy
   ignore heartbeat-retry-failure
   ignore heartbeat-recovery-timestamp-change
exit
 instance instance-id instance_id
   endpoint pfcp
     replicas replica_count
     sx-path-failure sx-detection-policy policy
     interface sxa
     sx-path-failure sx-detection-policy policy
     end
```

Configuring GTPC Path Failure Customization

To configure the GTPC path failure customization, use the following configuration:

```
config
   policy path-failure-detection policy_name
     max-remote-rc-change maximum_remote
     ignore echo-rc-change
     ignore control-rc-change
     ignore echo-failure
   exit
 exit
 instance instance-id instance_id
   endpoint gtp
     replicas replica_count
     vip-ip ipv4_address vip-port ipv4_port_number
     vip-ipv6 ipv6_address vip-ipv6-port ipv6_port_number
     dual-stack-transport { true | false }
     path-failure detection-policy policy
     interface [ s11 | s5e ]
     end
```

NOTES:

- When GTPC path failure detection policy isn't configured at interface-level, endpoint-level path failure detection policy is applicable.
- The max-remote-rc-change configuration specifies the counter change after which the S11 or S5 detects a peer restart. A peer restart is detected only if the absolute difference between the new and old restart counter is less than the value configured. For example, if the max-remote-rc-change is 10 and current peer restart counter is 251, then eGTP detects a peer restart only if the new restart counter is 252 through 255 or 0 through 5. Similarly, if the stored restart counter is 1, eGTP detects a peer restart only if the new restart counter is 2 through 11.
- Valid settings are from 1 to 255. The recommended setting is 32.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

GTPC Path Failure

Maintain statistics indicating number of times path failure was detected due to restart counter change in echo request or response message or control request or response message.

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="Local",data_center="DC",
gtpc_msg_type="gtpc_false_peer_restart_cfg_ctrl_rc_change",gtpc_peer_ip=
"209.165.201.17",instance_id="0",interface_type="S11",service_name="nodemgr"} 1
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="Local",data_center="DC",
gtpc_msg_type="gtpc_false_peer_restart_cfg_echo_rc_change",gtpc_peer_ip=
"209.165.201.17",instance_id="0",interface_type="S11",service_name="nodemgr"} 1
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="Local",data_center="DC",
gtpc_msg_type="gtpc_false_peer_restart_cfg_echo_rc_change",gtpc_peer_ip=
"209.165.201.27",instance_id="0",interface_type="S5E",service_name="nodemgr"} 1
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="Local",data_center="DC",
gtpc_msg_type="gtpc_ignore_echo_timeout",gtpc_peer_ip="209.165.201.27",
instance_id="0",interface_type="S5E",service_name="nodemgr"} 1
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="Local",data_center="DC",
gtpc_msg_type="gtpc_false_peer_restart_ignore_echo_rc_cfg",gtpc_peer_ip=
"209.165.201.27",instance_id="0",interface_type="S5E",service_name="nodemgr"} 1
```

```
nodemgr_gtpc_msg_stats{app_name="smf",cluster="Local",data_center="DC",
gtpc_msg_type="gtpc_false_peer_restart_ignore_ctrl_rc_cfg",gtpc_peer_ip=
"209.165.201.27",instance_id="0",interface_type="S5E",service_name="nodemgr"} 1
```

Table 92: GTPC Path Failure Statistics Descriptions

Statistics	Description
gtpc_false_peer_restart_cfg_echo_rc_change	The number of GTPC path failures ignored because Echo Restart Counter Change isn't within max-remote-rc-change configured.
gtpc_false_peer_restart_ignore_echo_rc_cfg	The number of GTPC path failures ignored because of Echo Restart Counter Change.
gtpc_false_peer_restart_cfg_ctrl_rc_change	The number of GTPC path failures ignored because Control Message Restart Counter Change isn't within max-remote-rc-change configured.
gtpc_false_peer_restart_ignore_ctrl_rc_cfg	The number of GTPC path failures ignored because of Control message Restart Counter Change.
gtpc_ignore_echo_timeout	The number of GTPC path failures ignored because of Echo Request timeout.

Sx Path Failure

Maintain statistics indicating number of times path failure was detected due to recovery timestamp change in the following messages.

```
nodemgr_up_pathfail_reasons{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",service_name="nodemgr",up_pathfail_reason="up_pathfail_ignored_hb_retry"}
 1
```

```
nodemgr_up_pathfail_reasons{app_name="smf",cluster="cn",data_center="cn",
instance_id="1",service_name="nodemgr",up_pathfail_reason="up_pathfail_ignored_hb_rt_change"}
 1
```

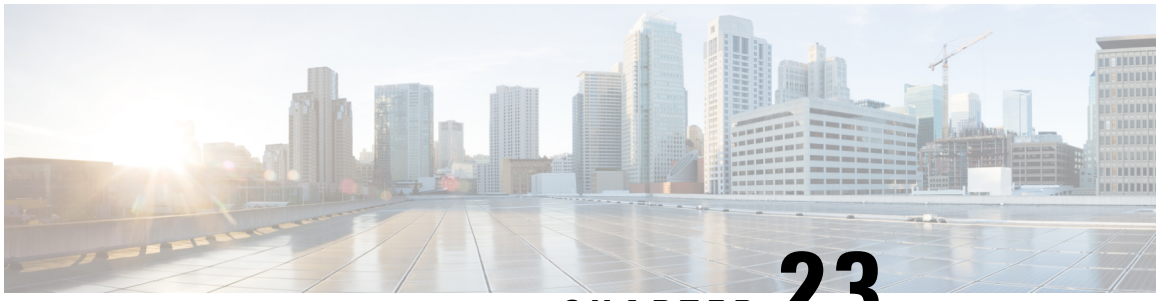
```
nodemgr_up_pathfail_reasons{app_name="smf",cluster="cn",data_center="cn",
instance_id="1",service_name="nodemgr",up_pathfail_reason="up_pathfail_reason_association_release"}
 1
```

```
nodemgr_up_pathfail_reasons{app_name="smf",cluster="cn",data_center="cn",
instance_id="1",service_name="nodemgr",up_pathfail_reason="up_pathfail_reason_hb_retry"}
 8
```

```
nodemgr_up_pathfail_reasons{app_name="smf",cluster="cn",data_center="cn",
instance_id="1",service_name="nodemgr",up_pathfail_reason="up_pathfail_reason_hb_rt_change"}
 1
```

Table 93: Sx Path Failure Statistics Descriptions

Statistics	Description
up_pathfail_ignored_hb_retry	The number of Sx path failures ignored because of Heartbeat Request timeout.
up_pathfail_reason_hb_retry	The number of Sx path failures detected because of Heartbeat Request timeout.
up_pathfail_ignored_hb_rt_change	The number of Sx path failures ignored because of Heartbeat Request Recovery Timestamp Change Ignored.
up_pathfail_reason_hb_rt_change	The number of Sx path failures detected because of Heartbeat Request Recovery Timestamp Change.
up_pathfail_reason_association_release	The number of Sx path failures detected because of Sx Association Release.



CHAPTER 23

GTPU Error Indication

- Feature Summary and Revision History, on page 243
- Feature Description, on page 244
- How it Works, on page 244
- Feature Configuration, on page 261
- OAM Support, on page 262

Feature Summary and Revision History

Summary Data

Table 94: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 95: Revision History

Revision Details	Release
Support added for signal peer (error indication and configuration is signal peer)	2021.02.3
First introduced.	2021.02.1

Feature Description

cnSGW-C supports the UPF reported GTPU errors in Session Report Request. UPF reports different GTPU errors to CP (cnSGW-C) in PFCP Session Report Request message.

cnSGW-C supports the following report requests.

- Error Indication Support (ERIR)
- Graceful Termination (GTER)
- Session Replacement (SRIR)

How it Works

This section describes how this feature works.

Error Indication Support

When cnSGW-C receives Error Indication with PFCP Session Report Request from UPF, it responds with PFCP Session Report Response and performs as per the configuration.

For cnSGW-C, signaling is based on configuration.

- S1U - local purge or page-ue
- S5U - local purge or peer signaling

Table 96: Error Indication Support (ERIR) Report Type

Interface	Configuration	TEID	Action
S1U	Local Purge	Default	Send SxSessionDeleteRequest to clean up on UPF Purge locally
		Dedicated	Send SxModReq (Remove Traffic Endpoint) Purge locally
		IDFT	Send SxMod (Remove IDFT Traffic Endpoint) - async Purge Bearer locally
	Page-UE	Default / Dedicated	Move UE to Idle state Send Sx_Modification _Request (Set FAR Action=BUFFER)
S5U	Local Purge	Default	Send SxDeleteSession Purge locally
		Dedicated	Send SxMod (Remove Traffic Endpoint) Purge Bearer locally
	Signal Peer	Default	Send SxMod (Drop) DBR/DSR, SxDelete
		Dedicated	Send DBR/DBC (Async), SxMod (Remove Traffic Endpoint)

Default Bearer with s1u as local-purge Call Flow

This section describes Default Bearer with s1u as local-purge call flow.

Figure 45: Default Bearer with s1u as local-purge Call Flow

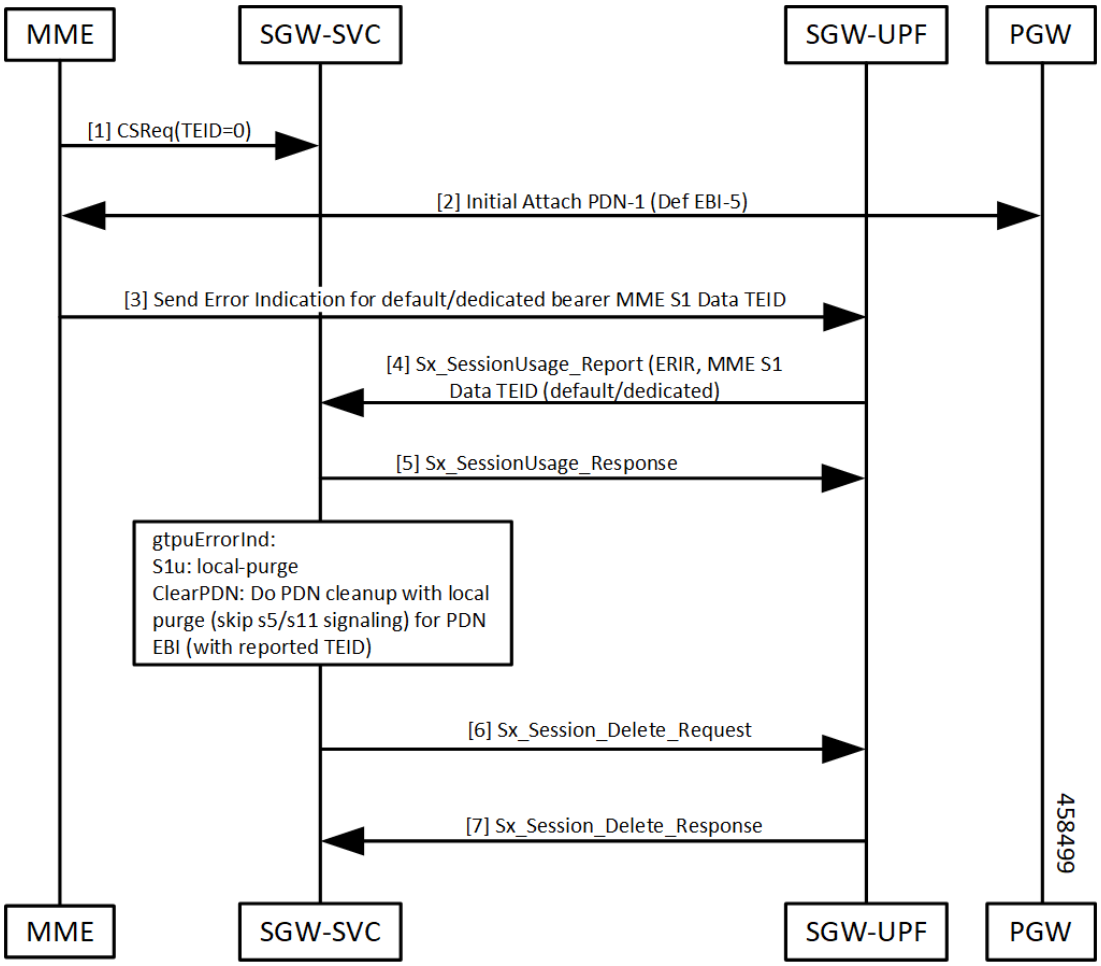


Table 97: Default Bearer with s1u as local-purge Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4, 5	• UPF sends Sx_Session_Report_Request with report type as ERIR and s1u TEID. • cnSGW-C responds with Sx_Session_Report_Response.

Step	Description
6, 7	cnSGW-C processes Sx_Session_Report_Request. gtpuErrorInd: s1u: local-purge If TEID received is for default bearer, submit internal transaction (T2) to clean up bearer (No peer signaling). • Send Sx_Session_Deletion_Request. • UPF responds with Sx_Session_Deletion_Response.

Dedicated Bearer with s1u as local-purge Call Flow

This section describes Dedicated Bearer with s1u as local-purge call flow.

Figure 46: Dedicated Bearer with s1u as local-purge Call Flow

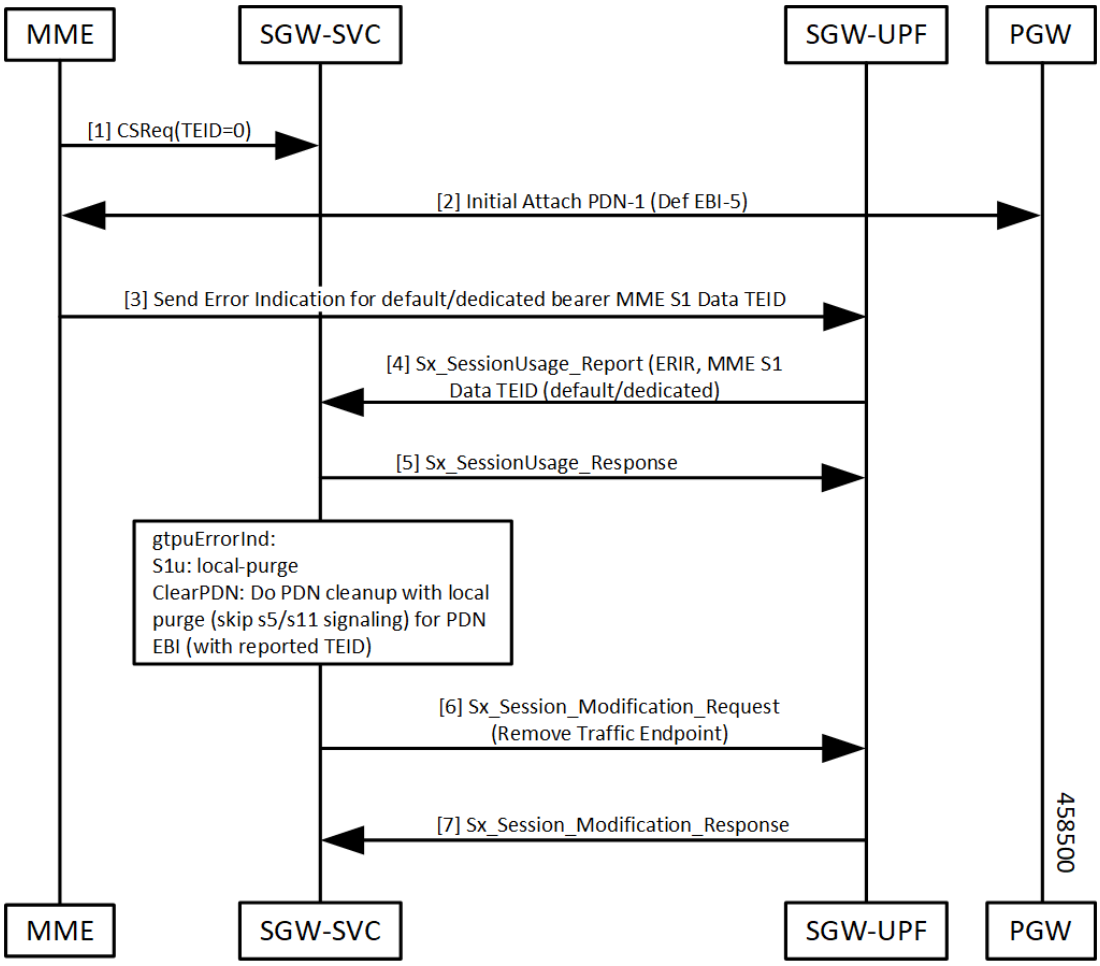


Table 98: Dedicated Bearer with s1u as local-purge Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4, 5	• UPF sends Sx_Session_Report_Request with report type as ERIR and s1u TEID. • cnSGW-C responds with Sx_Session_Report_Response.
6, 7	cnSGW-C processes Sx_Session_Report_Request. gtpuErrorInd: s1u: local-purge If TEID received is for dedicated bearer, submit internal transaction (T2) to clean up bearer (No peer signaling). • Send Sx_Session_Modification_Request (Remove Traffic Endpoint). • UPF responds with Sx_Session_Modification_Response.

Dedicated Bearer (IDFT) with s1u as local-purge Call Flow

This section describes Dedicated Bearer (IDFT) with s1u as local-purge call flow.

Figure 47: Dedicated Bearer (IDFT) with s1u as local-purge Call Flow

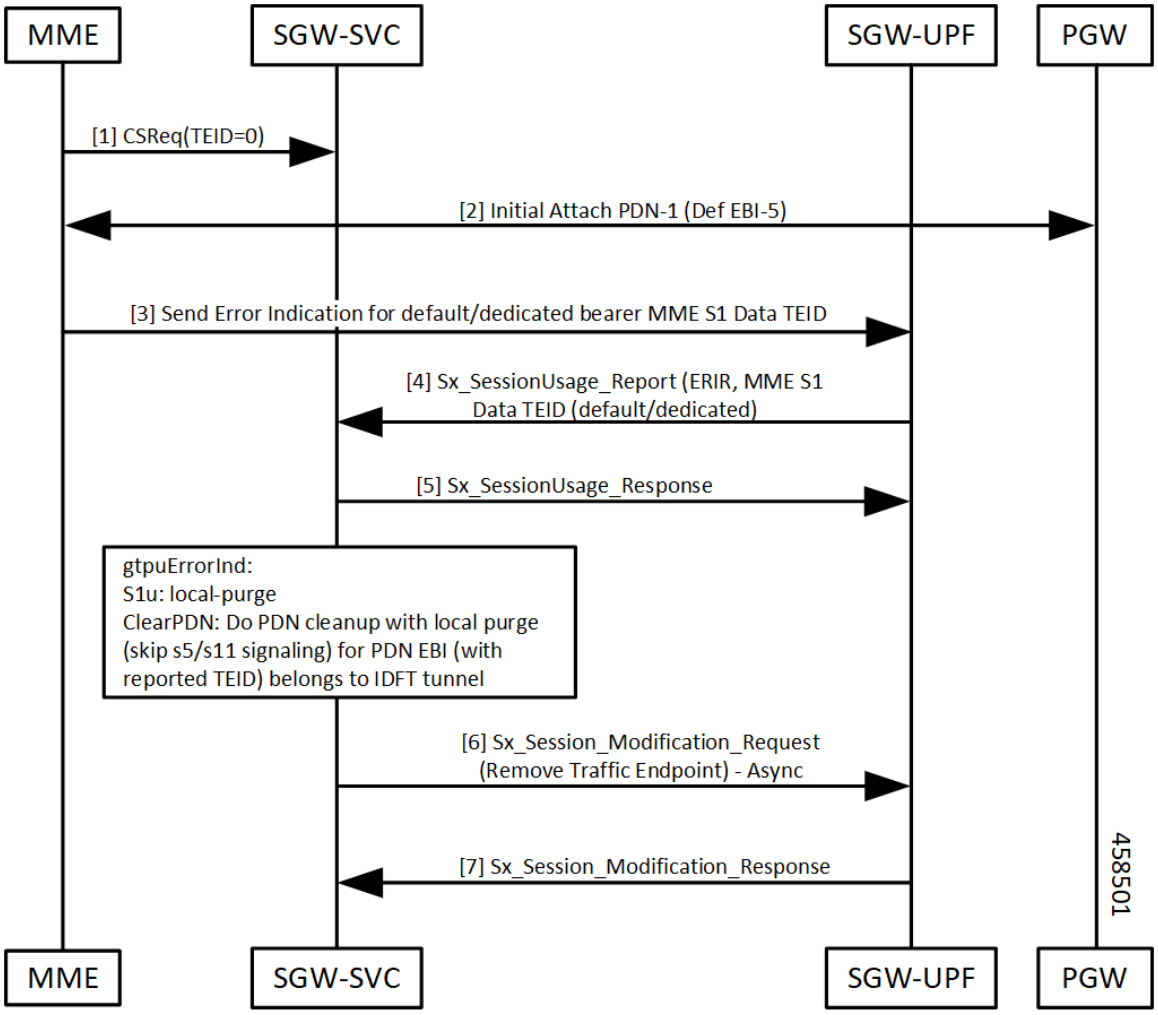


Table 99: Dedicated Bearer (IDFT) with s1u as local-purge Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4, 5	• UPF sends Sx_Session_Report_Request with report type as ERIR and s1u TEID. • cnSGW-C responds with Sx_Session_Report_Response.

Step	Description
6, 7	cnSGW-C processes Sx_Session_Report_Request. gtpuErrorInd: s1u: local-purge If TEID received is for dedicated bearer (IDFT), submit internal transaction (T2) to clean up bearer (No peer signaling). • Send Sx_Session_Modification_Request (Remove Traffic Endpoint). • UPF responds with Sx_Session_Modification_Response.

Default/Dedicated Bearer with s1u as page-ue Call Flow

This section describes Default/Dedicated Bearer with s1u as page-ue call flow.

Figure 48: Default/Dedicated Bearer with s1u as page-ue Call Flow

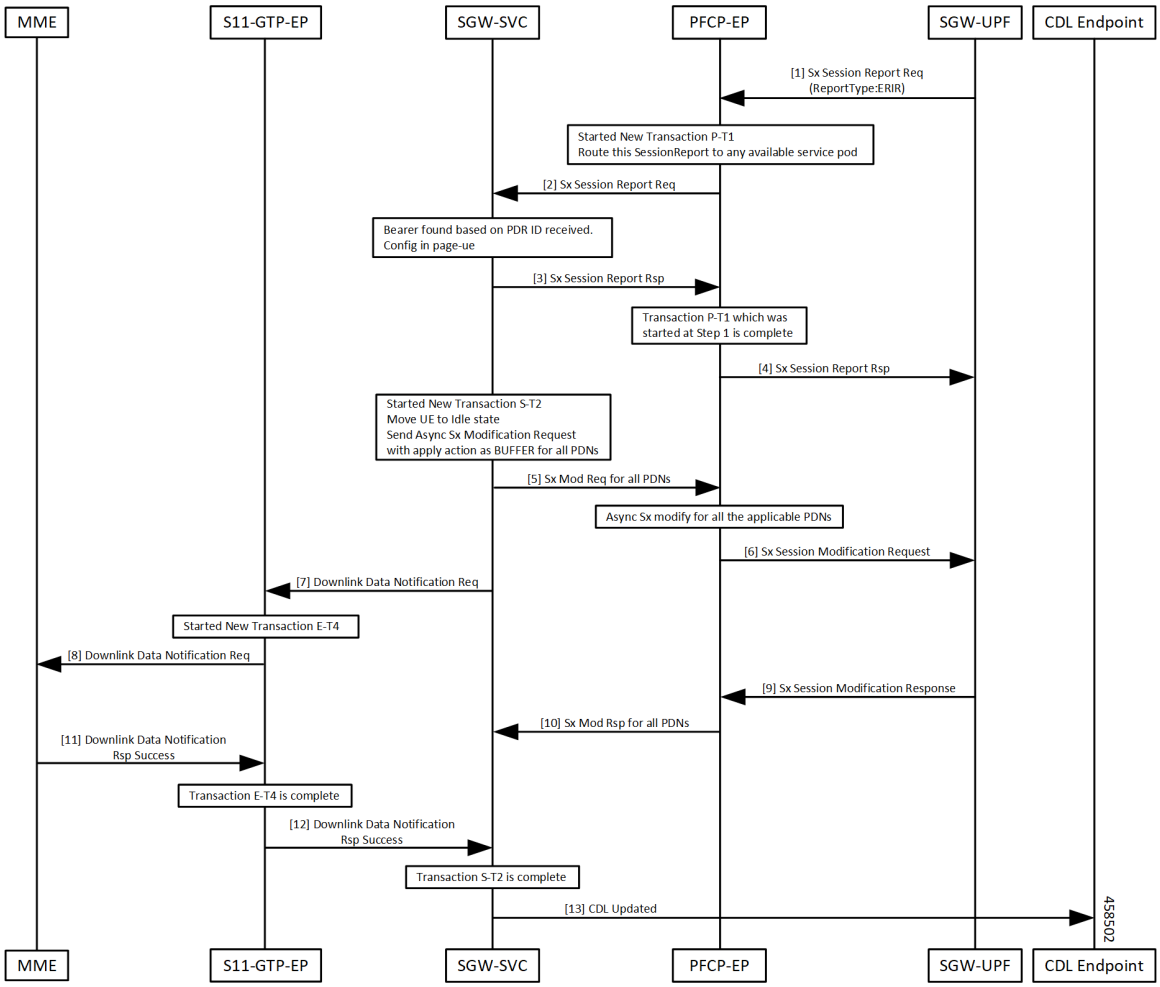


Table 100: Default/Dedicated Bearer with s1u as page-ue Call Flow Description

Step	Description
1, 2	SGW-UP sends Sx Session Report Req with type as ERIR to SGW service POD.
3, 4	SGW service POD sends Sx Session Report Res to SGW-UPF.
5, 6	PFCP-EP sends Sx Mod Req for all PDNs to SGW service POD. PFCP-EP sends Sx Session Modification Req to SGW-UP.
7, 8	SGW service POD sends Downlink Data Notification Req to S11-GTP-EP. S11-GTP-EP forwards Downlink Data Notification to MME.
9, 10	SGW-UP sends Sx session Modification Rsp to SGW service POD.
11, 12, 13	MME sends Downlink Data Notification Rsp Success to S11-GTP-EP. S11-GTP-EP forwards Downlink Data Notification Rsp Success to SGW service POD. SGW service POD sends CDL update to CDL endpoint when S-T2 transaction gets completed.

Default Bearer with s5u as local-purge/signal-peer Call Flow

This section describes Default Bearer with s5u as local-purge/signal-peer call flow.

Figure 49: Default Bearer with s5u as local-purge/signal-peer Call Flow

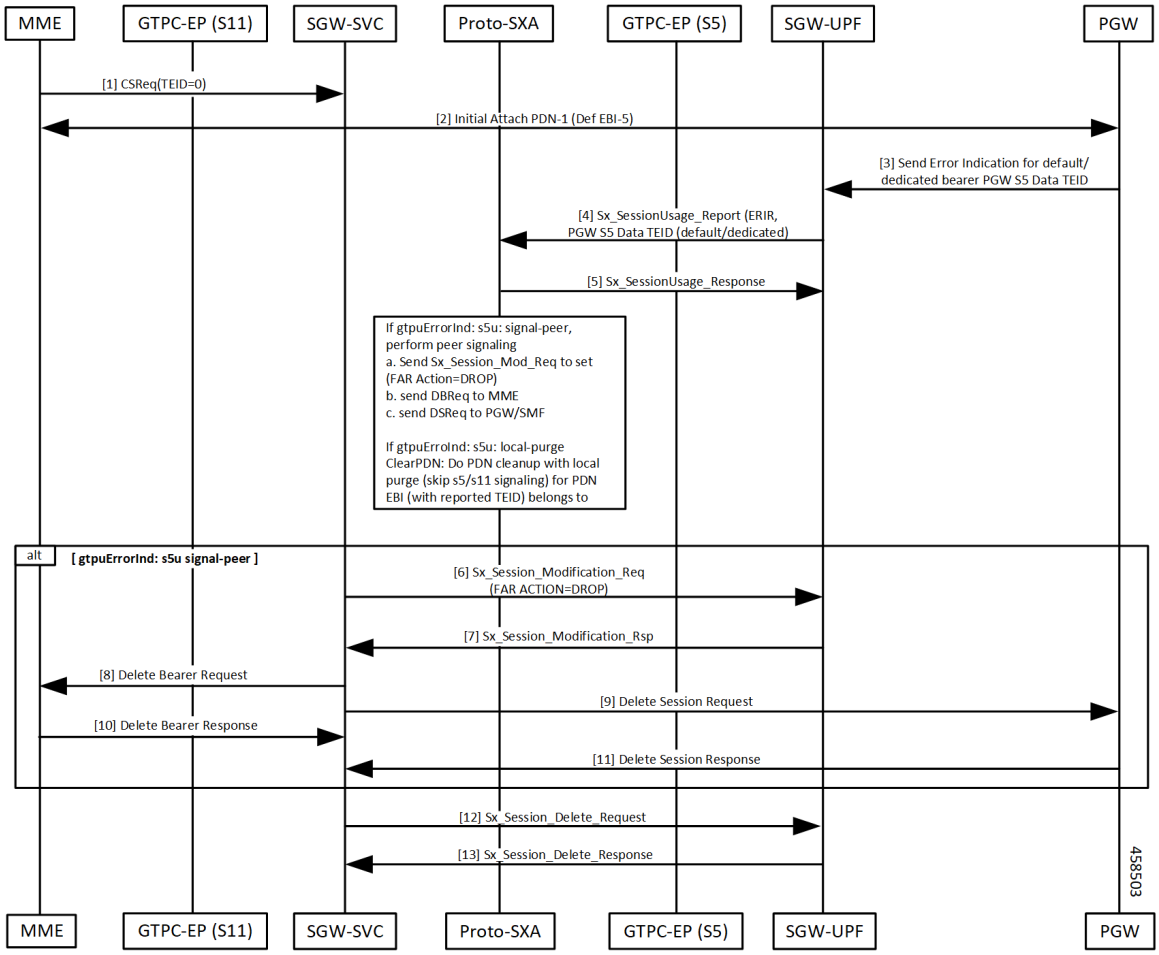


Table 101: Default Bearer with s5u as local-purge/signal-peer Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4, 5	GTPU Error detected on UPF. • Sx_Session_Report_Request sent to cnSGW-C. • cnSGW-C responds with Sx_Session_Resport_Response.

Step	Description
6-11	cnSGW-C processes Session Report Request (ERIR). If TEID is received for default bearer, submit internal transaction to clean up PDN (behavior depends on CLI configured). CLI: sgw-profile config If gtpuErrorInd: s5u: signal-peer • Send Sx_Session_Report_Request to UPF to set (FAR ACTION=DROP). • Send Delete Bearer Req to MME. • Send Delete Session Request to PGW.
12	Send Sx_Session_Delete_Request to UPF.
13	UPF responds with Sx_Session_Delete_Response.

Dedicated Bearer with s5u as local-purge/signal-peer Call Flow

This section describes Dedicated Bearer with s5u as local-purge/signal-peer call flow.

Figure 50: Dedicated Bearer with s5u as local-purge/signal-peer Call Flow

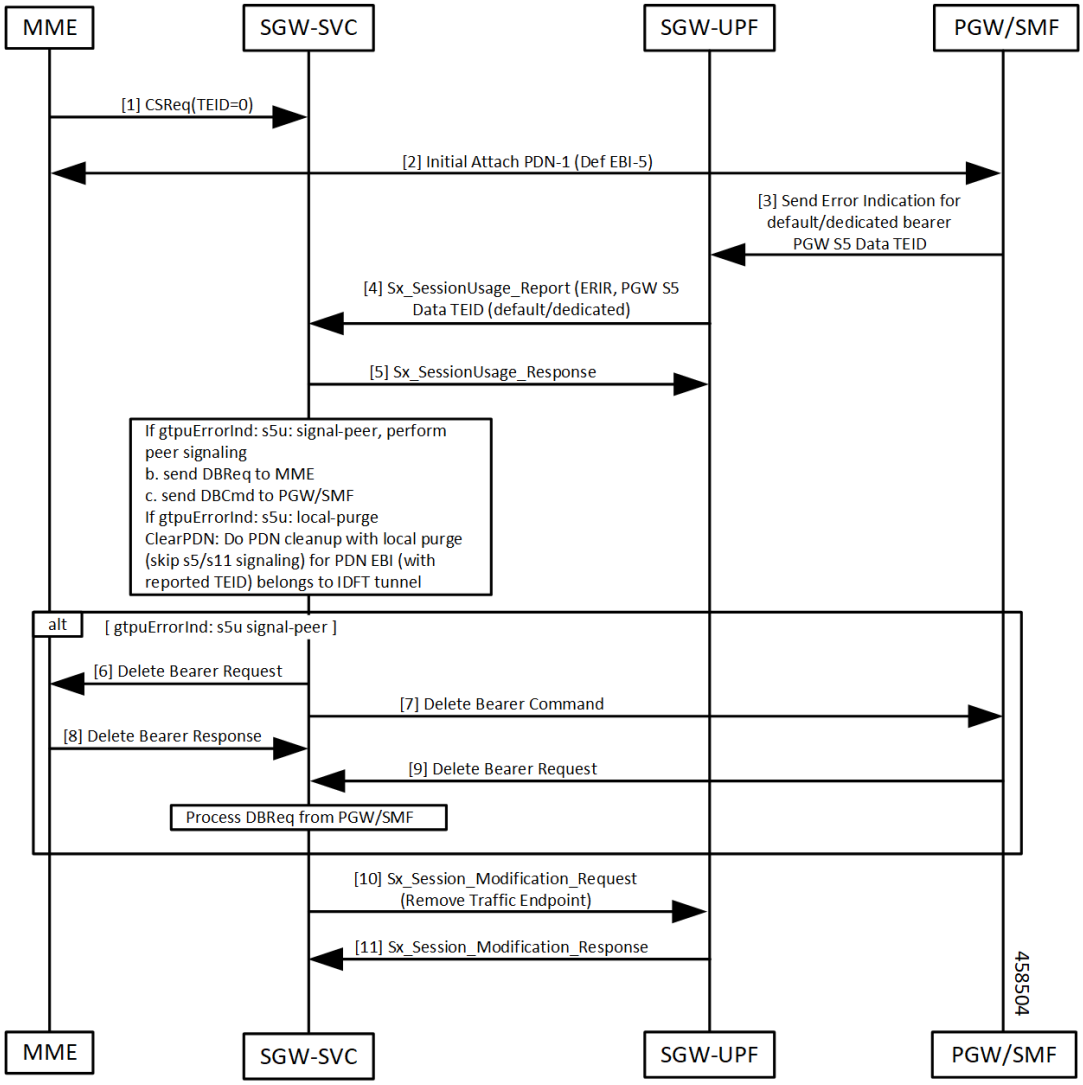


Table 102: Dedicated Bearer with s5u as local-purge/signal-peer Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4, 5	GTPU Error detected on UPF. • Sx_Session_Report_Request sent to cnSGW-C. • cnSGW-C responds with Sx_Session_Report_Response.

Step	Description
6, 7, 8, 9	cnSGW-C processes Session Report Request (ERIR). If TEID is received for dedicated bearer (s5u), submit internal transaction to clean up bearer. CLI: sgw-profile config If gtpuErrorInd: s5u: signal-peer • Send Delete Bearer Req to MME. • Send Delete Bearer Command to PGW.
10	Send Sx_Session_Modification_Request (Remove Traffic Endpoint) to UPF.
11	UPF responds with Sx_Session_Modification_Response.

Graceful Termination

When UPF can't recover PDU session during SR/ICSR recovery, it sends PFCP session Report Request to cnSGW with type as Graceful Termination Report (GTER).

When UPF can't load session during session recovery, it sends a GTER indicating to clear up all the interfaces for this reported session.

Graceful Termination Call Flow

This section describes Graceful Termination call flow.

Figure 51: Graceful Termination Call Flow

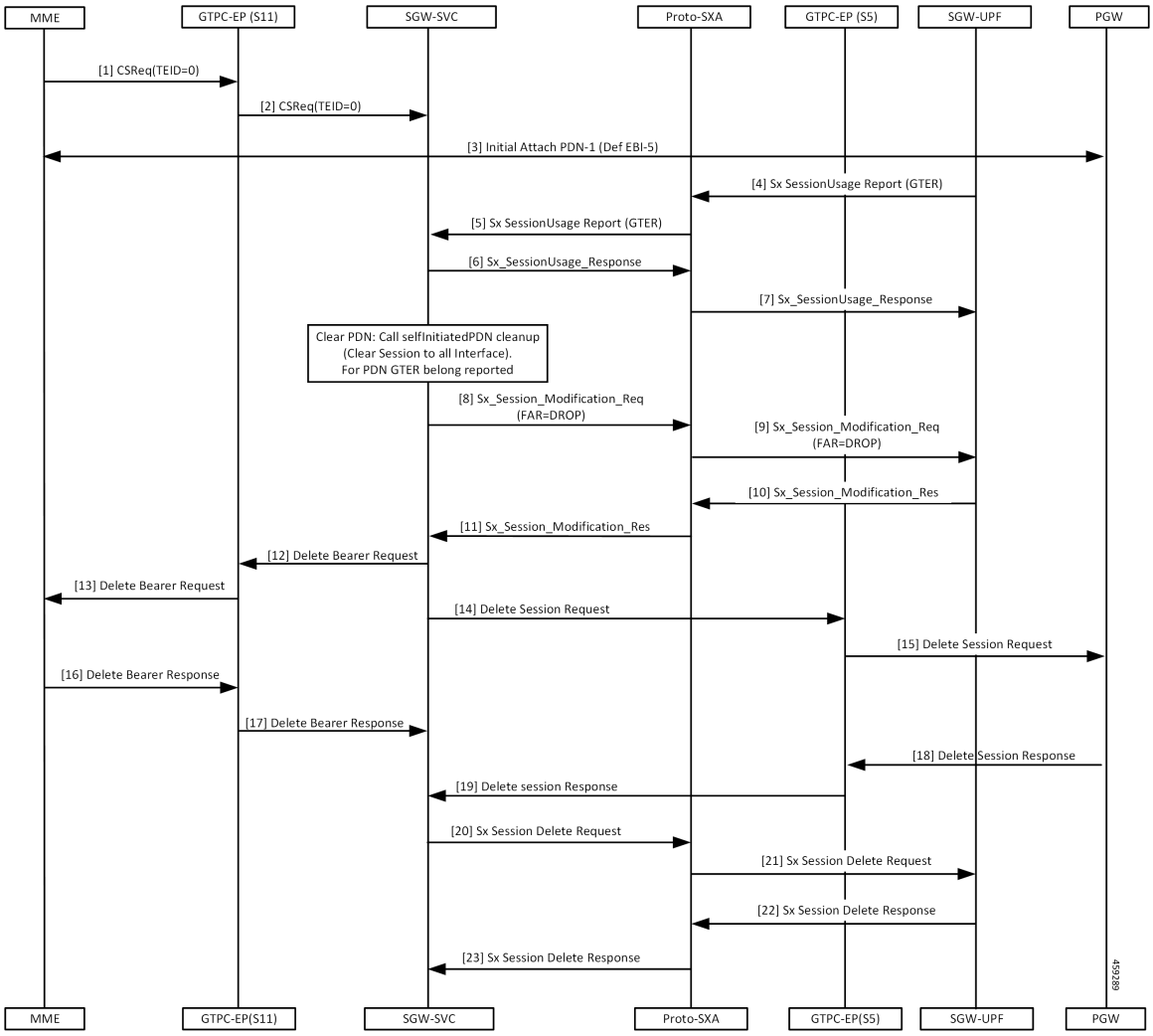


Table 103: Graceful Termination Call Flow Description

Step	Description
1, 2, 3	Initial attach complete.
4-7	• UPF sends Sx_Session_Report_Request with report type as GTER and TEID. • cnSGW-C responds with Sx_Session_Report_Response.
8-11	cnSGW-C processes Sx_Session_Report_Request and submits internal transaction (T2) to clean up PDN. • cnSGW-C sends Sx_Session_Modification_Request to set FAR Action=Drop. • UPF responds with Sx_Session_Modification_Response.

Step	Description
12-19	• Send Delete Bearer Req to MME. • Send Delete Session Request to PGW.
20-23	• Send Sx_Session_Delete_Request. • UPF responds with Sx_Session_Delete_Response.

Session Replacement

A Session Replacement (SRIR) is required when peer allocates same GTP-U TEID.

UPF sends SRIR report indicating to delete old session with same TEID. cnSGW-C uses GTPU path failure configuration for SRIR request processing.

Table 104: Session Replacement

Interface	Configuration	TEID	Action
S1U/S5U	Local Purge	Default	Send SxDeleteSession Purge locally
		Dedicated	Send SxMod (Remove Traffic Endpoint) Purge Bearer locally
	Signal Peer	Default	Send SxMod (Drop) DBR/DSR SxDelete
		Dedicated	Send DBR/DBC (Async) SxMod (Remove Traffic Endpoint)
IDFT	NA	NA	Send SxMod (Remove IDFT Traffic Endpoint)-async Purge Bearer locally

Session Replacement for Default Bearer Call Flow

This section describes the Session Replacement (SRIR) for Default Bearer call flow.

Figure 52: Session Replacement for Default Bearer Call Flow

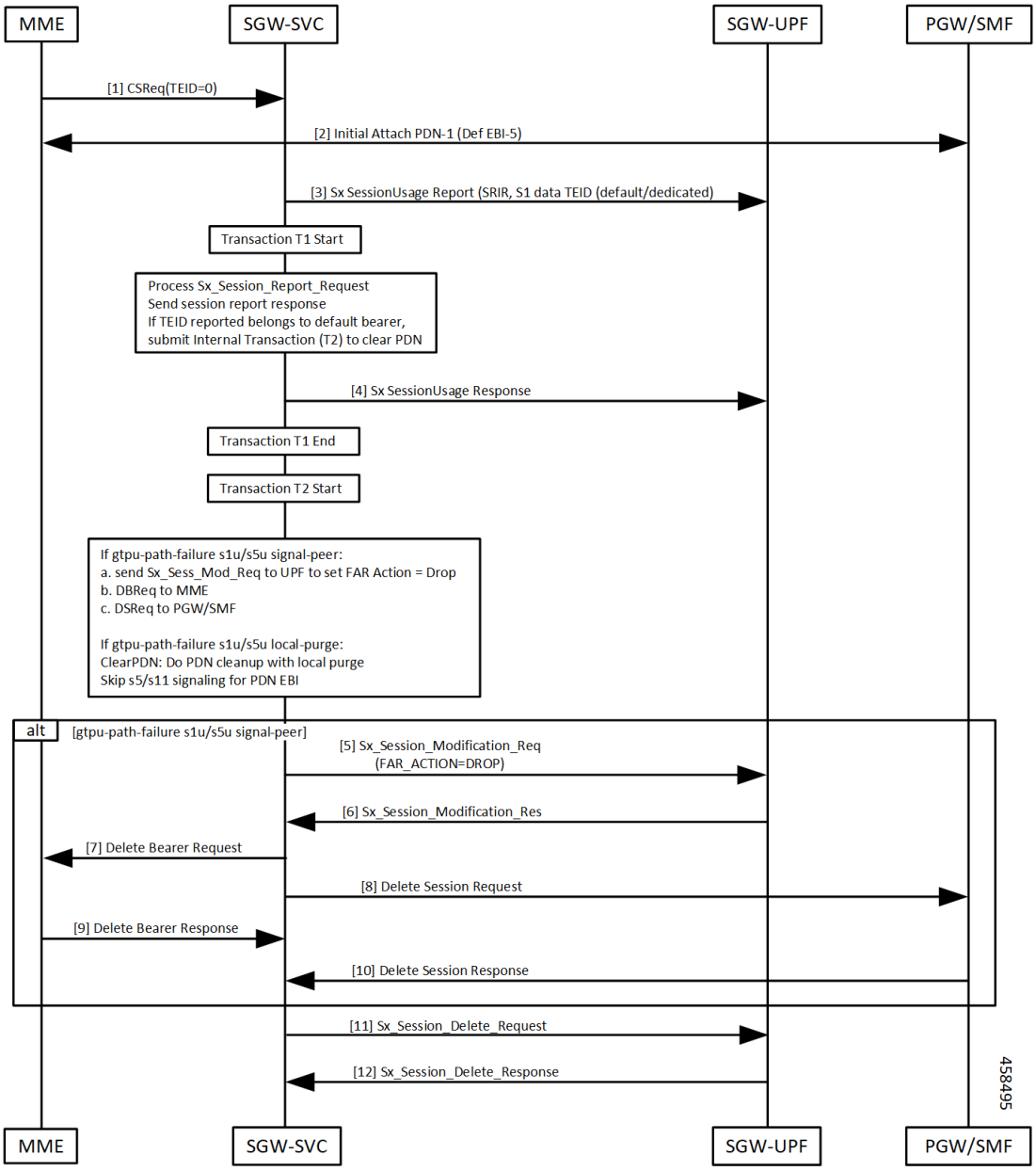


Table 105: Session Replacement for Default Bearer Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4	• UPF sends Sx_Session_Report_Request with report type as SRIR and TEID. • cnSGW-C responds with Sx_Session_Report_Response.

Step	Description
5–10	cnSGW-C processes Sx_Session_Report_Request, wrong Session Replacement uses GTPU path failure CLI for peer-signaling or local purge. If TEID is received for default bearer, submit internal transaction (T2) to clean up PDN. If CLI, `gtpu-path-failure s1u/s5u signal-peer` • Send Sx_Session_Report_Request to UPF to set FAR ACTION=DROP. • Send Delete Bearer Req to MME. • Send Delete Session Request to PGW.
11, 12	• Send Sx_Session_Delete_Request • UPF responds with Sx_Session_Delete_Response.

Session Replacement for Dedicated Bearer Call Flow

This section describes the Session Replacement (SRIR) for Dedicated Bearer call flow.

Figure 53: Session Replacement for Dedicated Bearer Call Flow

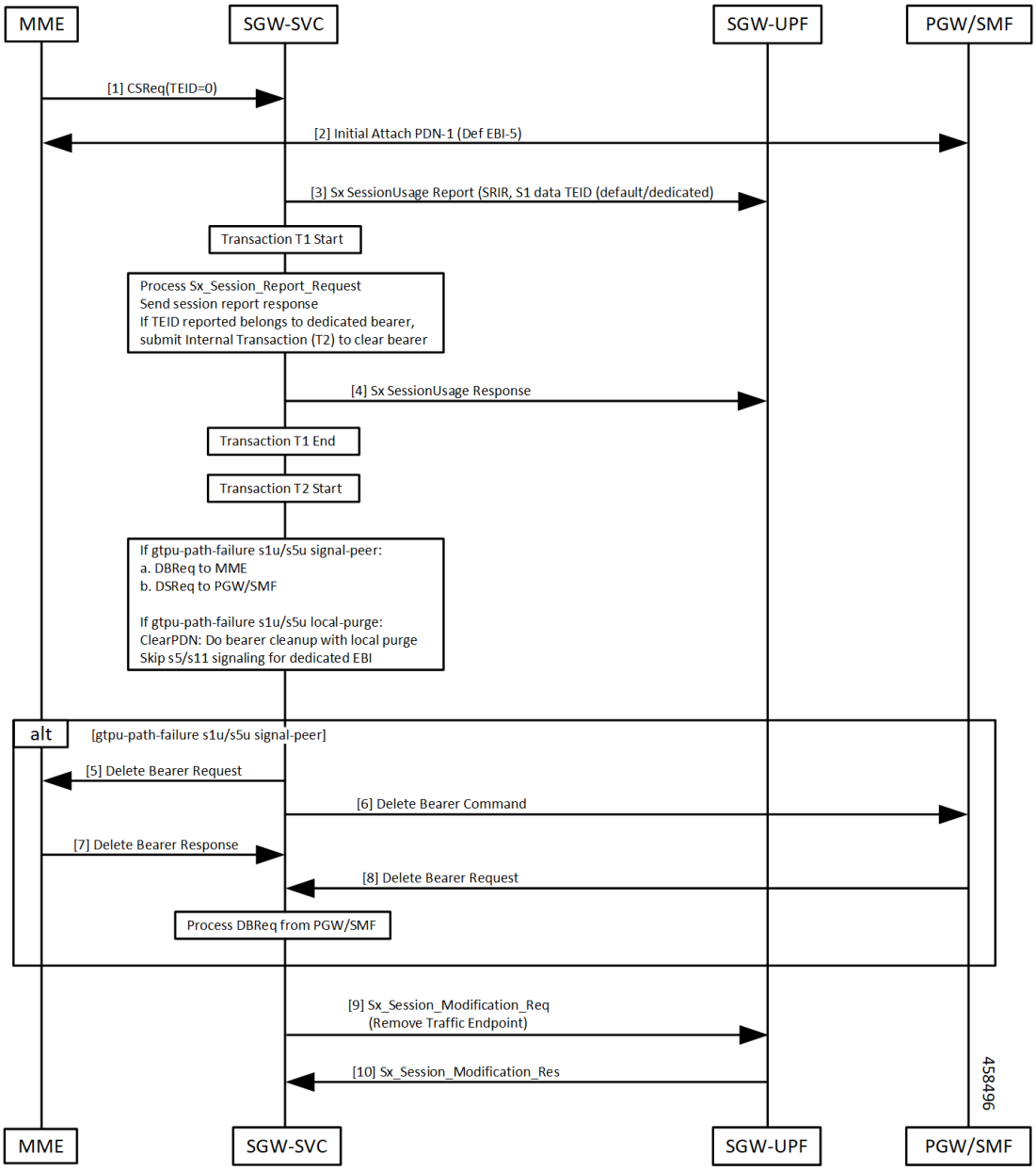


Table 106: Session Replacement for Dedicated Bearer Call Flow Description

Step	Description
1, 2	Initial attach complete.
3, 4	• UPF sends Sx_Session_Report_Request with report type as SRIR and TEID. • cnSGW-C responds with Sx_Session_Report_Response.

Step	Description
5–10	cnSGW-C processes Sx_Session_Report_Request, wrong Session Replacement uses GTPU path failure CLI for peer-signaling or local purge. If TEID is received for dedicated bearer, submit internal transaction (T2) to clean up bearer. If CLI, `gtpu-path-failure s1u/s5u signal-peer` • Send Sx_Session_Report_Request to UPF to set FAR ACTION=DROP. • Send Delete Bearer Req to MME. • Send Delete Session Request to PGW.
11, 12	• Send Sx_Session_Modification_Request (Remove Traffic Endpoint) • UPF responds with Sx_Session_Modification_Response.

Feature Configuration

This section describes how to configure the GTPU Error Indication feature.

To configure this feature, use the following configuration:

```
config
 profile sgw sgw_profile_name
  gtpu-error-ind
   s1u [ local-purge | page-ue ]
   s5u [ local-purge | signal-peer ]
   end
```

NOTES:

- **s1u**—S1-U interface.
- **s5u**—S5-U interface.
- **local-purge**—Locally purge the affected bearers or PDNs without informing peer.
- **page-ue**—Reset to S1-Idle state and initiate paging for this UE.
- **signal-peer**—Clear the affected bearers or PDNs with signaling towards peer.

Configuration Example

The following is an example configuration.

```
config
profile sgw sgw1
gtpu-error-ind s1u local-purge
gtpu-error-ind s5u signal-peer s1u local-purge
end
```

Conifguration Verification

To verify the configuration:

```
show running-config profile sgw gtpu-error-ind s1u local-purge
profile sgw sgw1
gtpu-error-ind s1u local-purge
```

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

The following are statistics for PDN cleanup due to Error Report.

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="Local",data_center="DC",instance_id="0",
pdn_type="ipv4",rat_type="EUTRAN",reason="s1u_gtpu_error",service_name="sgw-service"}
 1
```

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="Local",data_center="DC",instance_id="0",
pdn_type="ipv4",rat_type="EUTRAN",reason="s5u_gtpu_error",service_name="sgw-service"}
 1
```

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="Local",data_center="DC",instance_id="0",
pdn_type="ipv4",rat_type="EUTRAN",reason="s1u_gtpu_session_replacement",service_name="sgw-service"}
 1
```

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="Local",data_center="DC",instance_id="0",
pdn_type="ipv4",rat_type="EUTRAN",reason="userplane_requested_graceful_termination",service_name="sgw-service"}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="s5u_gtpu_error_initiated_bearer_deletion",status="attempted",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="s5u_gtpu_error_initiated_bearer_deletion",status="success",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="s5u_gtpu_session_replacement_initiated_bearer_deletion",status="attempted",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_egress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"s5u_gtpu_session_replacement_initiated_bearer_deletion",status="success",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"s5u_gtpu_error_initiated_bearer_deletion",status="attempted",sub_fail_reason=""}
 1
```

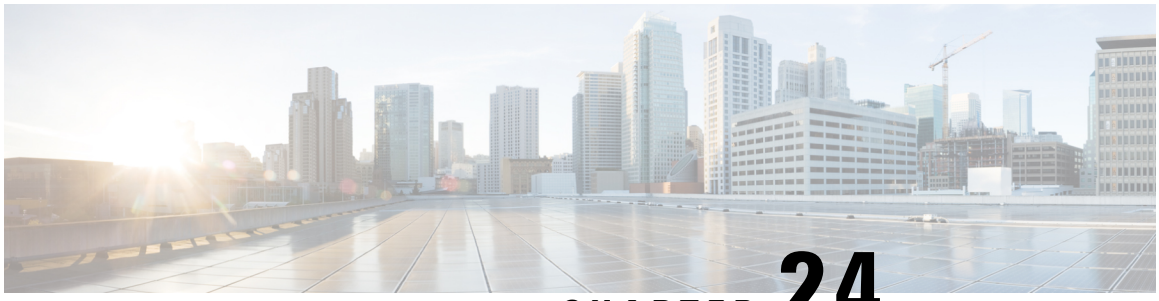
```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"s5u_gtpu_error_initiated_bearer_deletion",status="success",sub_fail_reason=""}
1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"s5u_gtpu_session_replacement_initiated_bearer_deletion",status="attempted",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",gr_instance_id="1",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"s5u_gtpu_session_replacement_initiated_bearer_deletion",status="success",sub_fail_reason=""}
 1
```

The following SGW ddn_stats_type is added for DDN initiated due to GTPU Error indication on S1u tunnel.

```
sgw_ddn_stats{app_name="smf",cluster="cn",data_center="cn",ddn_stats_type="gtpu_err_ind_triggered",
instance_id="0",service_name="sgw-service"} 2
```



CHAPTER 24

GTPU Path Failure

- Feature Summary and Revision History, on page 265
- Feature Description, on page 266
- How it Works, on page 266
- Feature Configuration, on page 270
- GTPU Path Failure OAM Support, on page 271

Feature Summary and Revision History

Summary Data

Table 107: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	

Revision History

Table 108: Revision History

Revision Details	Release
Support to clean session or bearer based on reported value in node-report (Node-ID and Peer Information)	2021.02.3
First introduced.	2021.02.1

Feature Description

When UPF detects a GTP-U path failure, it sends Node Report Request (with NodeID and GTPU Peer Information) to cnSGW-C. cnSGW-C clears the PDU sessions belonging to the GTP-U peer and UPF node ID.

This feature supports the following:

- Sending Node Report Success
- Cleaning session or bearer based on the reported value in node-report (Node-ID and Peer Information)
- Incrementing the relevant statistics

How it Works

This section describes how this feature works.

The following table describes various actions on detecting GTPU path failure.

Table 109: GTPU Path Failure for Node Report

Interface	Configuration	TEID	Action
s1u/s5u	Local Purge	Default	Send SxSessionDeletion to clean up on UPF Purge PDN locally
		Dedicated	Send SxSessionModification (Remove TrafficEndpoint) Purge Bearer locally
	Signal Peer	Default	Send SxSessionModificationRequest (FAR Action=DROP) Send DBReq to MME and DSReq to PGW Send SxSessionDeletionRequest
		Dedicated	Send DBReq to MME and DBCmd to PGW (Async) Send SxSessionModificationRequest (Remove Traffic Endpoint)

Call Flows

This section describes key call flows of GTPU Path Failure.

Path Failure for Default Bearer Call Flow

This section describes the Path Failure for Default Bearer call flow.

Figure 54: Path Failure for Default Bearer Call Flow

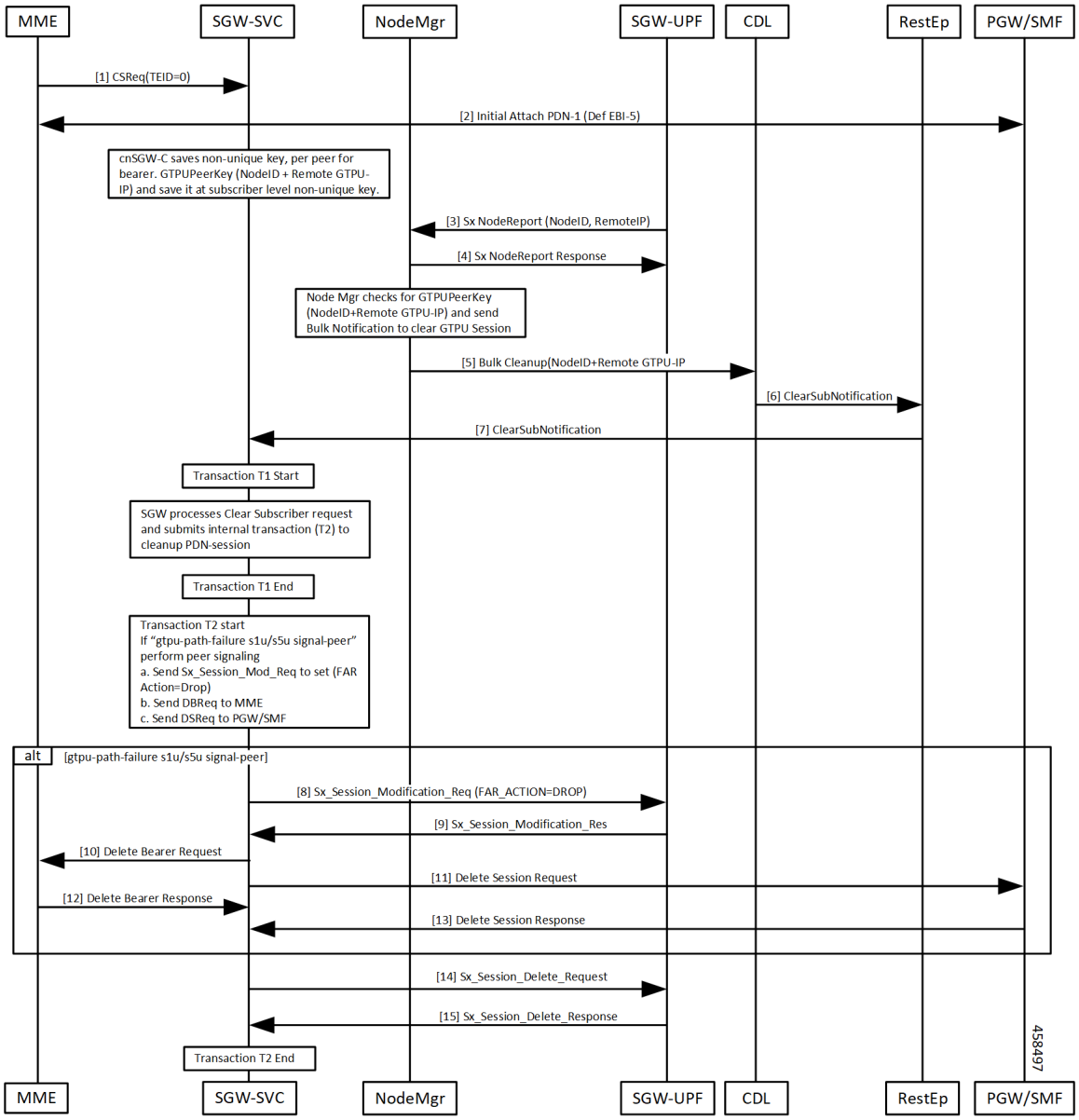


Table 110: Path Failure for Default Bearer Call Flow Description

Step	Description
1, 2	Initial attach complete. cnSGW-C saves non-unique GTPUPeerKey (NodeID:Remote GTPU-peer-IP) per bearer.
3, 4	• Path failure detected on UPF. UPF sends NodeReportRequest to Node Manager. • Node Manager responds with NodeReportResponse. • Initiate Bulk Cleanup request to CDL.
6, 7	• CDL sends ClearSubNotification to RestEp. • RestEP forwards it to cnSGW-C.
8–13	cnSGW-C processes Clear Subscriber Request. If GTPU peer IP received is for default bearer, submit internal transaction (T2) to clean up PDN. If CLI `gtpu-path-failure s1u/s5u signal-peer` • Send Sx_Session_Report_Request to UPF to set FAR ACTION=DROP. • Send Delete Bearer Req to MME. • Send Delete Session Request to PGW.
14, 15	• Send Sx_Session_Delete_Request. • UPF responds with Sx_Session_Delete_Response.

Path Failure for Dedicated Bearer Call Flow

This section describes the Path Failure for Dedicated Bearer call flow.

Figure 55: Path Failure for Dedicated Bearer Call Flow



Table 111: Path Failure for Dedicated Bearer Call Flow Description

Step	Description
1, 2	Initial attach complete. cnSGW-C saves non-unique GTPUPeerKey (NodeID:Remote GTPU-peer-IP) per bearer.
3, 4	• Path failure detected on UPF. UPF sends NodeReportRequest to Node Manager. • Node Manager responds with NodeReportResponse. • Initiate Bulk Cleanup request to CDL.

Step	Description
6, 7	• CDL sends ClearSubNotification to RestEp. • RestEP forwards it to cnSGW-C.
8–13	cnSGW-C processes Clear Subscriber Request. If GTPU peer IP received is for dedicated bearer, submit internal transaction (T2) to clean up PDN. If CLI `gtpu-path-failure s1u/s5u signal-peer` • Send Delete Bearer Req to MME. • Send Delete Bearer Command to PGW.
14, 15	• Send Sx_Session_Modification_Request (Remove Traffic Endpoint). • UPF responds with Sx_Session_Delete_Response.

Feature Configuration

This section describes how to configure the GTPU Path Failure feature.

To configure this feature, use the following configuration.

```
config
 profile sgw sgw_profile_name
  gtpu-path-failure
   s1u [ local-purge | signal-peer ]
   s5u [ local-purge | signal-peer ]
   end
```

NOTES:

- **s1u**—S1-U interface. Default is local-purge.
- **s5u**—S5-U interface. Default is local-purge.
- **local-purge**—Locally purge the affected bearers or PDNs without informing peer.
- **signal-peer**—Clear the affected bearers or PDNs with signaling towards peer.

Configuration Example

The following is an example configuration.

```
config
   profile sgw sgw1
   gtpu-path-failure s1u local-purge
   gtpu-path-failure s5u local-purge
   end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw sgw1
profile sgw sgw1
sgw-charging-threshold thre1
sgw-charging-profile ch1
locality LOC1
fqdn cisco.com.apn.epc.mnc456.mcc123
charging-mode gtpp
subscriber-policy sub1
session-idle-timer 86000
ddn failure-action-drop-timer 60
ddn no-user-connect-retry-timer 60
path-failure s11 signal-peer
path-failure s5e signal-peer
gtpu-error-ind s5u signal-peer
gtpu-path-failure s1u local-purge
gtpu-path-failure s5u local-purge
```

GTPU Path Failure OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

- Node Manager Statistics

```
nodemgr_node_report_stats{app_name="smf",backlog_tmr="0",cluster="Local",data_center="DC",
instance_id="0",node_report_no_of_sess="0",node_report_peer_gtpu="209.165.201.20:209.165.201.30",
node_report_type="",service_name="nodemgr",session_tmr="0",status="attempted",
up_ep_key="209.165.201.20:209.165.201.10"} 1
```

```
nodemgr_node_report_stats{app_name="smf",backlog_tmr="1617268831815934340",cluster="Local",
data_center="DC",instance_id="0",node_report_no_of_sess="0",
node_report_peer_gtpu="209.165.201.20:209.165.201.30",node_report_type="origin",
service_name="nodemgr",session_tmr="600",status="success",
up_ep_key="209.165.201.20:209.165.201.10"} 1
```

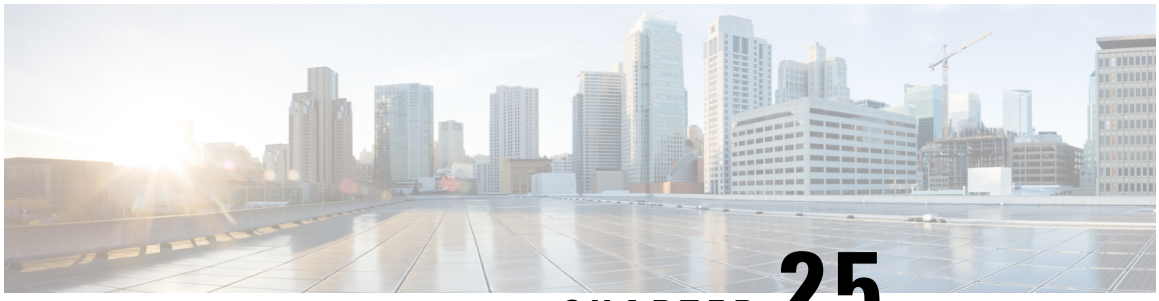
- SGW Service Statistics

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_egress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="s5u_gtpu_path_failure_initiated",
status="attempted",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_egress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="s5u_gtpu_path_failure_initiated",
status="success",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_ingress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="s5u_gtpu_path_failure_initiated",
status="attempted",sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_ingress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="s5u_gtpu_path_failure_initiated",
status="success",sub_fail_reason=""} 1
```



CHAPTER 25

GTPv2 and Sx Messages Retransmission and Timeout Handling

- Feature Summary and Revision History, on page 273
- Feature Description, on page 274
- How it Works, on page 274
- Configuring the Retransmission and Timeout Values, on page 275

Feature Summary and Revision History

Summary Data

Table 112: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 113: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

cnSGW-C enables the retransmission and timeout handling for the parameters associated with outbound and inbound messages through CLI. The retransmission and timeout handling is applicable for the:

- Messages over a GTPC interface towards MME and PGW and
- Sx messages sent towards the User Plane (UP).



Note For handling the retransmission and timeout parameters, you must add the retransmission configuration (N3/T3) for the interface (S5e, S11, and Sxa).

How it Works

S-GW service, GTPC-EP, and SMF protocol are the primary nodes involved in the the retransmission and timeout handling.

The SGW-serice is responsible for:

- Handling the timeout event from GTPC-EP and SMF protocol
- Ignoring the inbound retransmitted message

The GTPC-EP and SMF protocol is responsible for:

- Retransmission and timeout handling
- Reading the N3/T3 configuration
- Updating the N3/T3 on configuration change

The retransmission and timeout handling is applicable for both outbound and inbound messages.

Outbound Message

To supports retransmission and timeout of outgoing GTP and PFCP messages, you must configure an interface specific N3 (maximum number of retries) and T3 (retransmission timeout) timer values in accordance to network response time/delay time.

The MME/S11 peers can have different retransmission timeout as compared to PGW/S5 or UPF/SXA.

The GTPC-EP/Protocol pod retries the outgoing request messages based on configured N3T3 values until the response is received or N3T3 is exhausted. In case of N3T3 gets exhausted, the GTPC-EP/Protocol pod sends the failure response with cause peer no response to service pod to indicate that no response has been received for outgoing request message.

Inbound Message

At each N4 and GTP endpoint, there's a set of queues for incoming and outgoing traffic. Each queue has a dispatcher thread running that pulls the message from the queue. It dispatches the message to the application for further processing.

Each dispatcher references a retransmission cache to check if the incoming request is already in service. It further performs the following actions:

- If it's a retry request, the dispatcher drops the incoming request.
- If the retransmission cache reaches the threshold for outstanding requests, the incoming request is dropped.

Each dispatcher has a separate retransmission cache. This cache is also updated with the response of the request sent. It's for the retransmission request received after the response is sent.

Configuring the Retransmission and Timeout Values

This section includes the CLI commands to configure the retransmission and timeout values for the outbound and inbound messages.

Following is the CLI configuration for the outbound messages:

```
config
  instance instance-id instance_id
    endpoint endpoint_name
      interface interface_name
        retransmission timeout timeout_intervalmax-retry retry_value
        end
```

NOTES:

- **instance instance-id** *instance_id*—Specify the instance ID.
- **endpoint** *endpoint_name*—Specify the endpoint name.
- **interface** *interface_name*—Specify the interface name.
- **retransmission timeout** *timeout_interval*—Configure the timeout interval value.

Following is the CLI configuration for the inbound messages:

```
config
    instance instance-id 1
    endpoint protocol
    interface n4
        dispatcher
            count 5
            outbound true
        threshold 5000
        end
```

NOTES:

- **capacity** *capacity_value*—Specify the queue size for each dispatcher queue. The default value is 5000.
- **count** *value*—Specify the number of supported dispatcher queues for the interface or the endpoint.

- **expiry** *expiry_duration*—Specify the duration for which the cache entry with response is held in the cache. The default value is 60 seconds.
- **nonresponsive** *nonresponsive_duration*—Specify the duration for which the cache entry without response is held in the cache.
- **outbound** *true | false*—Disable dispatcher queue support for outgoing messages. The default value is true. When set to false, the queue support is enabled for outgoing messages.

  It means by default, the queue support is enabled for the outgoing messages. Must be one of the following:

  - *true*—Disable dispatcher queue support for outgoing messages, set the **outbound** to true.
  - *false*—Enable dispatcher queue support for outgoing messages, set the **outbound** to false.

- **rate-limit** *rate_limit*—Specify the rate limit for each queue.
- **threshold** *threshold*—Specify the outstanding limit for non-responsive cache entries. When the threshold is reached, the incoming requests are dropped. It must be an integer. The default value is 30000 milliseconds.

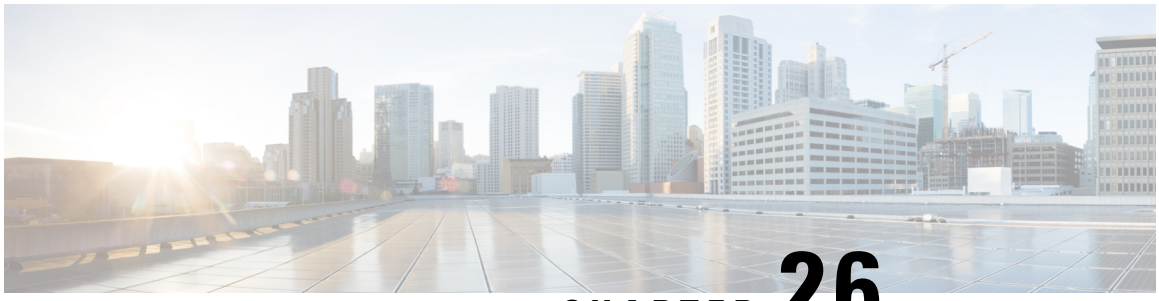
Configuration Verification

Following is the sample configuration to verify the retrnasmission and timeout handling configuration for the outbound and inbound messages:

```
show running-config instance instance-id 1 endpoint gtp
instance instance-id 1
endpoint gtp
replicas 1
interface s5e
retransmission timeout 2 max-retry 2
sla response 7000
dispatcher
count 1
capacity 1000
outbound true
threshold 10000
expiry 40000
nonresponsive 20000
exit
vip-ip 209.165.201.25
exit
interface s11
retransmission timeout 2 max-retry 2
sla response 7000
dispatcher
count 1
capacity 1000
outbound true
threshold 10000
expiry 40000
nonresponsive 20000
exit
vip-ip 209.165.201.2
exit
exit

show running-config instance instance-id 1 endpoint pfcp
instance instance-id 1
```

```
endpoint pfcp
replicas 1
interface sxa
retransmission timeout 2 max-retry 2
dispatcher
count 1
capacity 1000
outbound true
threshold 10000
expiry 40000
nonresponsive 20000
exit
heartbeat
interval 0
retransmission-timeout 3
max-retransmissions 5
exit
retransmission timeout 5 max-retry 1
exit
interface n4
heartbeat
interval 0
retransmission-timeout 3
max-retransmissions 5
exit
exit
exit
```



CHAPTER 26

GTPv2 Load/Overload Support

- Feature Summary and Revision History, on page 279
- Feature Description, on page 279
- Configuring the GTPv2 Load and Overload Feature, on page 281
- GTPv2 Load and Overload OAM Support, on page 289

Feature Summary and Revision History

Summary Data

Table 114: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 115: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

The following are the details for the load control and the overload control features.

Load Control

The load control enables a GTPC entity, such as SGW or PGW, to send its load information to a GTPC peer, such as MME, ePDG, and TWAN. This information is used to balance the session load across all the nodes supporting the same function, such as SGW cluster, as per their effective loads. The load information reflects the operational status of the GTPC entity resources.

cnSGW-C load control behavior is as follows:

- Activate or deactivate the load control support in cnSGW-C, using the CLI.
- When the load control feature is activated, cnSGW-C signals its load control information to the MME for the following reasons:
  - Optimum GW selection procedures
  - Enhanced load balancing across cnSGW-C in the network
- The calculation of the load control information is based on the deployment scenarios.
- The applicable GTPC request or the response message contains piggybacked load control information.
- cnSGW-C includes only the single instance of LCI (Load Control Information) IE as per the SGW load control information. cnSGW-C sends the LCI IE received from the PGW to the MME, along with its LCI information.
- The frequency of load control information inclusion at SGW is based on the deployment scenario. The SGW ensures the propagation of the new or the updated load control information to the target receivers is within the acceptable delay. This acceptable delay helps in achieving the effective load balancing act in the network.

Overload Control

The overload control enables a GTPC entity to reduce gracefully its own incoming signaling load by instructing its GTPC peers to send the reduced traffic. The GTPC entity reaches overload, when it operates above the signaling capacity. This overload results in a diminished performance, resulting in to impacts on the incoming and the outgoing traffic handling. The GTPC node uses the load information to reduce, or throttle, or reduce and throttle, the amount of GTPC signaling traffic between these nodes.

cnSGW-C overload control behavior is as follows:

- Activate or deactivate overload control support in cnSGW-C using the CLI.
- When the overload control feature is activated, cnSGW-C signals its overload control information to the MME or the PGW. This helps in controlling the GTPC signaling traffic towards itself.
- SGW supports the handling of the overload control information in all the applicable messages.
- The applicable GTPC request or the response message contains piggybacked overload control information.
- cnSGW-C includes only the single instance of OCI (Overload Control Information) IE as per the SGW overload control information. cnSGW-C sends the OCI IE received from the PGW to the MME, along with its OCI information.
- The calculation of the overload control information is based on the deployment scenario.
- cnSGW-C rejects with the cause as GTPC entity congestion, when the SGW is in self-protection mode.

- SGW doesn't store the MME or the PGW overload control information.
- SGW doesn't perform throttling towards the MME and the PGW.

Note The load control and the overload control are optional features.

Configuring the GTPv2 Load and Overload Feature

This section describes how to configure the GTPv2 load or overload conditions.

Configuring this feature involves the following steps:

- Configuring the Load Profile: This section describes how to configure the load profile and the parameters required to calculate the load of cnSGW-C.
- Configure the Overload Exclude Profile: This section describes how to make an exclusion and configure the exclude profile in overload conditions.
  - This profile determines the session-related messages to exclude from the throttling decisions.
  - Both self-protection and peer overload control, use this configuration.
- Configuring the Overload Condition Profile, on page 283: This section describes how to configure the profile in overload conditions.
  - The profile determines the various conditions for overload control and the resulting throttling decisions.
  - It supports only one overload profile.
  - The load profile supports overload profile functionality.
- Configuring the Maximum Session Count, on page 285: This section describes how to configure the maximum session count that contributes to the session percent load factor in LCI/OCI calculation.
- Associating the Overload-Profile with SGW-Profile Association, on page 285: The association of the Overload-Profile and the SGW-Profile, can be configured.
- Configure the Load Factor Frequency: This section describes the step-wise process to configure the load factor frequency.

Configuring the Load Profile

To configure this feature use the following configuration:

```
config
   profile load profile_name
      load-calc-frequency load_calc_frequency_value
      load-fetch-frequency load_fetch_frequency_value
      advertise
```

```
        interval interval_value
        change-factor change_factor_value
        exit
    interface gtpc
        action advertise
end
```

NOTES:

- **profile load** *profile_name*—Specify the load profile name.
- **load-calc-frequency** *load_calc_frequency_value*—Specify the system load calculation time in seconds. Must be an integer in the range of 5-3600. The default value is 10 seconds.
- **load-fetch-frequency** *load_fetch_frequency_value*—Specify the time interval in seconds at which protocol pods fetch load from the cache POD. Must be an integer in the range of 5-3600. The default value is 10 seconds.
- **advertise interval** *interval_value*—Specify the time interval of sending LCI to the peers in seconds. Must be an integer in the range of 0-3600. The default value is 300 seconds.
- **advertise change-factor** *change_factor_value*—Specify the LCI value to corresponding peers, if the difference between the current load value and the last indicated load value is greater than the change-factor. Must be an integer in the range of 1-20. The default value is five.
- **interface gtpc action advertise**—Enables LCI publishing on the GTPC interface.

Configuration Example

The following is an example configuration.

```
config
profile load pl
load-calc-frequency 30
load-fetch-frequency 60
advertise
interval 300
change-factor 1
exit
interface gtpc
action advertise
end
```

Configure the Overload Exclude Profile

In the self-protection mode, as a default behavior, cnSGWc throttles all the incoming GTPC messages. However, as the high-priority messages cannot be throttled, cnSGWc requires to exempt those high-priority messages even during the overload scenarios. Configuring the overload exclude profile allows you to define the interfaces from which the high-priority incoming messages should be exempted from throttling.

Before you begin

Configure the load profile.

Procedure

Step 1 Configure an overload exclude profile instance in the global configuration mode using the command **profile overload-exclude** *overload_exclude_profile_name*.

Example:

```
[sgw] smf#config
[sgw] smf(config)#profile overload-exclude excludeProf
[sgw] smf(config-overload-exclude-excludeProf)#
```

Step 2 Define one of the overload exclude profile parameters, that is, DNN, ARP, QCI and Message Priority, to exempt the messages from throttling, using the command **{ dnn-list** *dnn_list* **| arp-list** *arp_list* **| qci-list** *qci_list* **| message-priority { any | s5e | s11 | sxa } upto** *priority_val***}**.

Example:

```
[sgw] smf(config-overload-exclude-excludeProf)# dnn-list [ sos ]
[sgw] smf(config-overload-exclude-excludeProf)# arp-list [ 2 3 4 ]
[sgw] smf(config-overload-exclude-excludeProf)# qci-list [ 1 4 9 ]
[sgw] smf(config-overload-exclude-excludeProf)# message-priority s11 upto 5
```

What to do next

After configuring the overload exclude profile, you must perform the following tasks:

- Configure the overload condition profile.
- Configure the maximum session count.
- Associate the overload exclude profile with the cnSGWc profile.
- Configure the load factor frequency calculation.

Configuring the Overload Condition Profile

To configure this feature use the following configuration:

```
config
   profile overload overload_profile_name
      overload-exclude-profile self-protection self_protection_profile_name
         node-level
         tolerance
            minimum min_percentage
            maximum max_percentage
         reduction-metric
            minimum min_percentage
            maximum max_percentage
         interface gtpc
            overloaded-action advertise
               advertise
```

```
interval interval_value
change-factor change_factor_value
validity-period validity_period_value
end
```

NOTES:

- **profile overload** *overload_profile_name*—Specify the overload profile name.
- **overload-exclude-profile self-protection** *self_protection_profile_name*—(This is an optional configuration) Exclude messages from throttling decisions in self-protection condition.
- **tolerance minimum** *min_percentage* **maximum** *max_percentage*—Specify the system overload limits. Refer the following scenarios:
  - When the system load is less than *min_percentage*, the system is in a normal state.
  - When the system load is in between *min_percentage* and *max_percentage*, the system is in an overloaded state. In this scenario, the node overload control action is triggered.
  - When the system load is greater than *max_percentage*, the system is in a self-protection state. In this scenario, the self-protection action is triggered.
  - *max_percentage* must be an integer in the range of 1-100. The default value is 95.
  - *min_percentage* must be an integer in the range of 1-100. The default value is 80.
- **reduction-metric minimum** *min_percentage* **maximum** *max_percentage*—Specify the reduction metric limits. Refer the following scenarios:
  - Both percentage values, *min_percentage* and *max_percentage* work along with the **tolerance** configuration.
  - The percentage value *max_percentage* must be an integer in the range of 1-100. The default value is 100.
  - The percentage value *min_percentage* must be an integer in the range of 1-100. The default value is 10.

  Example: Send 10 percent OCI to peer nodes, when the load is 80 percent, and 30 percent, when the load is 95 percent, during the following conditions:
  - **tolerance** *min_percentage* is 80 and *max_percentage* is 95.
  - **reduction-metric** *min_percentage* is 10 and *max_percentage* is 30.
- **interface gtpc overloaded-action advertise**—Configures the action on GTPC interface when a node gets overloaded. GTPC includes S5/S8/S11/S2b interfaces. Certain actions apply only to specific interfaces.
- **advertise interval** *interval_value*—Specify the periodicity of sending LCI to the peers in seconds. Must be an integer in the range of 0-3600. The default value is 300 seconds.
- **advertise change-factor** *change_factor_value*—Specify the change-factor value. GTPC sends the LCI to corresponding peers, if the difference between the current load value and the lastly indicated load value is greater than the change-factor value. Must be an integer in the range of 1-20. The default value is five.

- **advertise validity-period** *validity_period_value*— Specify the validity period of the advertised OCI value in seconds. Must be an integer in the range of 1-3600. The default value is 600 seconds.

Configuring the Maximum Session Count

To configure this feature use the following configuration:

```
config
   profile converged-core profile_name
      max-session-count max_session_count_value
      end
```

NOTES:

- **profile converged-core** *profile_name*—Specify the name of the converged core profile.
- **max-session-count** *max_session_count_value*—Specify the maximum number of sessions supported. Must be an integer in the range of 1-12000000.

Configuration Example

The following is an example configuration.

```
config
profile converged-core convergedCoreProfile
max-session-count 12000000
exit
```

Associating the Overload-Profile with SGW-Profile Association

The association of the Overload-Profile and the SGW-Profile, can be configured.

To configure this feature use the following configuration:

```
config
   profile overload overload_profile_name
      overload-exclude-profile self-protection self_protection_profile_name
   node-level
      tolerance
         minimum min_percentage
         maximum max_percentage
      reduction-metric
         minimum min_percentage
         maximum max_percentage
         advertise
         interval interval_value
         change-factor
         exit
      interface gtpc
         overloaded-action [ advertise ]
         exit
      exit
   exit
```

```
    profile load load_name
    load-calc-frequency load_calc_frequency_value
    load-fetch-frequency load_fetch_frequency_value
    advertise
    interval interval_value
    change-factor change_factor_value
```

```
    exit
    interface gtpc
    action advertise
    exit
exit
profile sgw sgw_name
load-profile profile_name
overload-profile overload_profile_name
end
```

NOTES:

- **overload** *overload_name*—Specify the overload protection profile name. Must be a string.
- **overload-exclude-profile**—Excludes profiles for overload scenarios.
- **self-protection** *overload_value*—Specify the profile to be excluded for self-protection. Must be a string.
- **tolerance minimum** *min_percentage*—Specify the minimum tolerance level below which the system is in a normal state. Must be an integer in the range of 1-100. The default value is 80.
- **tolerance maximum** *max_percentage*—Specify the maximum tolerance level above which the system is in a self-protection state. Must be an integer in the range of 1-100. The default value is 95.
- **reduction-metric minimum** *min_percentage*—Specify the percentage of reduction along with minimum tolerance-level for configuration. Must be an integer in the range of 1-100. The default value is 10.
- **reduction-metric maximum** *max_percentage*—Specify the percentage of reduction along with maximum tolerance-level for configuration. Must be an integer in the range of 1-100. The default value is 100.
- **interval** *interval_value*—Specify the advertising interval in seconds. Must be an integer in the range of 0-3600. The default value is 300 seconds.
- **validity** *validity_value*—Specify the validity period of the advertised OCI value in seconds. Must be an integer in the range of 1-3600. The default value is 600 seconds.
- **change-factor** *change_factor_value*—Specify the minimum change between current OCI and last indicated OCI, after which the advertising should happen. Must be an integer in the range of 1-20. The default value is five.
- **profile load** *load_name*—Specify the name of the load profile. Must be a string.
- **load-calc-frequency** *load_calc_frequency_value*—Specify the system load calculation interval in seconds. Must be an integer in the range of 5-3600. The default value is 10 seconds.

- **load-fetch-frequency** *load_fetch_frequency_value*—Specify the time interval in seconds at which the service pods fetch load from the cache pod. Must be an integer in the range of 5-3600. The default value is 10 seconds.
- **load-profile** *profile_name*—Specify the name of the load profile.
- **overload-profile** *overload_profile_name*—Specify the name of the overload profile.
- : Specify the exclude overload profile name:
  - : Specify the ARP list that needs to be excluded from throttling decisions. Must be an integer in the range of 1-15. Maximum eight entries are allowed.
  - : Specify the list of DNNs that needs to be excluded from throttling decision. Maximum three entries are allowed.
  - **message-priority**: Specify upto which message periority to be excluded from throttling decisions.
  - **procedure-list**: Procedures to be excluded from throttling decisions. This parameter is applicable only for Self-Protection.
  - : Specify the QoS Class Identifier to be excluded from throttling decisions. Must be an integer in the range of 1-.254. Maximum 8 entries are allowed. For example, range values can be 1-9,65,66,69,70,80,82,83,128-254.

Configuration Example

The following is an example configuration.

```
config
profile overload op
overload-exclude-profile self-protection <overload-exclude-profile-name>
node-level
tolerance minimum 5
tolerance maximum 50
reduction-metric minimum 50
reduction-metric maximum 100
advertise
interval 0
change-factor 1
exit
interface gtpc
overloaded-action [ advertise ]
exit
exit
exit
profile load lp
load-calc-frequency 120
load-fetch-frequency 15
advertise
interval 0
change-factor 1
exit
interface gtpc
action advertise
exit
exit
profile sgw <sgw_name>
load-profile <profile_name>
```

```
overload-profile <overload_profile_name>
end
```

Configuration Verification

To verify the configuration:

```
show running-config profile
profile sgw sgw1
load lp1
overload op1
end
```

Configure Load Factor Parameters

cnSGWc calculates the load factor of the system at specific time intervals and advertises it to the peer nodes. However, this configuration allows you to specify the load factor calculation frequency along with the non-participating pods to be excluded. If the load factor calculation is not configured, cnSGWc, by default, calculates the load factor at an interval of 30 seconds excluding all the app-infra pods.

Procedure

Step 1 Configure load factor frequency calculation using the command **load factor**.

Example:

```
[sgw] smf(config)# load factor
```

Step 2 Define the load factor calculation frequency or the app-infra pods to exclude using the command **[no] load factor { calc-frequency** *calc_frequency_time* **| exclude-pods** *exclude_pod_name***}**.

Example:

```
[sgw] smf(config)# load factor calc-frequency 15
[smf] smf(config)# load factor exclude-pods bgpspeaker-pod
```

- The default value of the CLI **calc-frequency** is 30.
- The CLI **exclude-pods** excludes following app-infra pods:
  - **bfdmgr**
  - **bgpspeaker-pod**
  - **cache-pod**
  - **edr-monitor**
  - **georeplication-pod**
- Both the CLIs **calc-frequency** and **exclude-pods** are backward-compatible.

Step 3 Exit the global configuration mode using the command **exit**.

Example:

```
[sgw] smf(config)#exit
```

Configuration Verification

The show command **show running-config load** displays the load factor configuration parameters.

```
[sgw] smf# show running-config load
  load factor calc-frequency 30
  load factor exclude-pods bgpspeaker-pod
exit
```

GTPv2 Load and Overload OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

Normal

```
node_overload_status{app_name="smf", cluster="cn", data_center="cn",
instance_id="0", service_name="oam-pod"} 0
```

Overload

```
node_overload_status{app_name="smf", cluster="cn", data_center="cn",
instance_id="0", service_name="oam-pod"} 1
```

Self-Protection

```
node_overload_status{app_name="smf", cluster="cn", data_center="cn",
instance_id="0", service_name="oam-pod"} 2
```

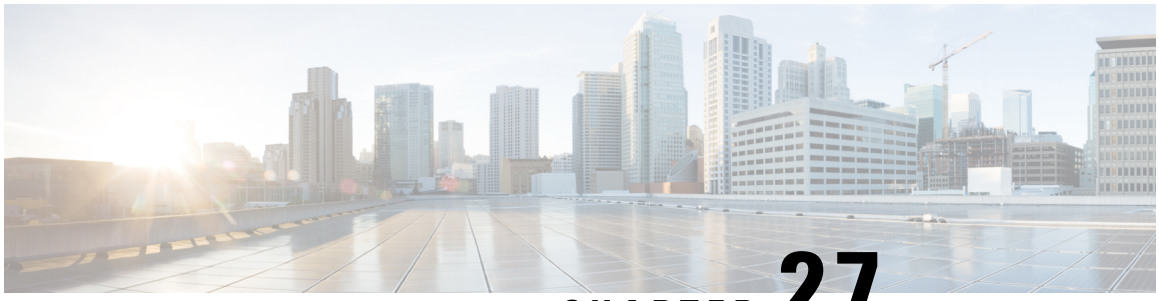
SGW Service Statistics

```
sgw_service_stats{app_name="smf", cluster="cn", data_center="cn",
fail_reason="gtp_entity_in_congestion", instance_id="0",
interface="interface_sgw_ingress", reject_cause="entity_in_congestion",
service_name="sgw-service", sgw_procedure_type="initial_attach",
status="rejected", sub_fail_reason=""}
```

LCI/OCI Metric Values

```
node_lci_metric{app_name="SGW", cluster="cn", component="oam-pod",
data_center="DC", namespace="cn", instance_id="0", service_name="oam-pod"}
```

```
node_oci_metric{app_name="SGW", cluster="cn", component="oam-pod",
data_center="DC", namespace="cn", instance_id="0", service_name="oam-pod"}
```



CHAPTER 27

GTPv2 Message Validation

- Feature Summary and Revision History, on page 291
- Feature Description, on page 291
- How it Works, on page 292

Feature Summary and Revision History

Summary Data

Table 116: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 117: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

cnSGW-C supports basic GTPv2 message validation of IEs (values, mandatory IE, and service-dependent IE), and sends responses from the SGW-Service/GTPC-EP pod.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Basic and Advance Validation on SGW-Ingress (S11) Call Flow

The following section describes the Basic and Advance Validation on SGW-Ingress (S11) call flow.

Figure 56: Basic and Advance Validation - S11 Call Flow

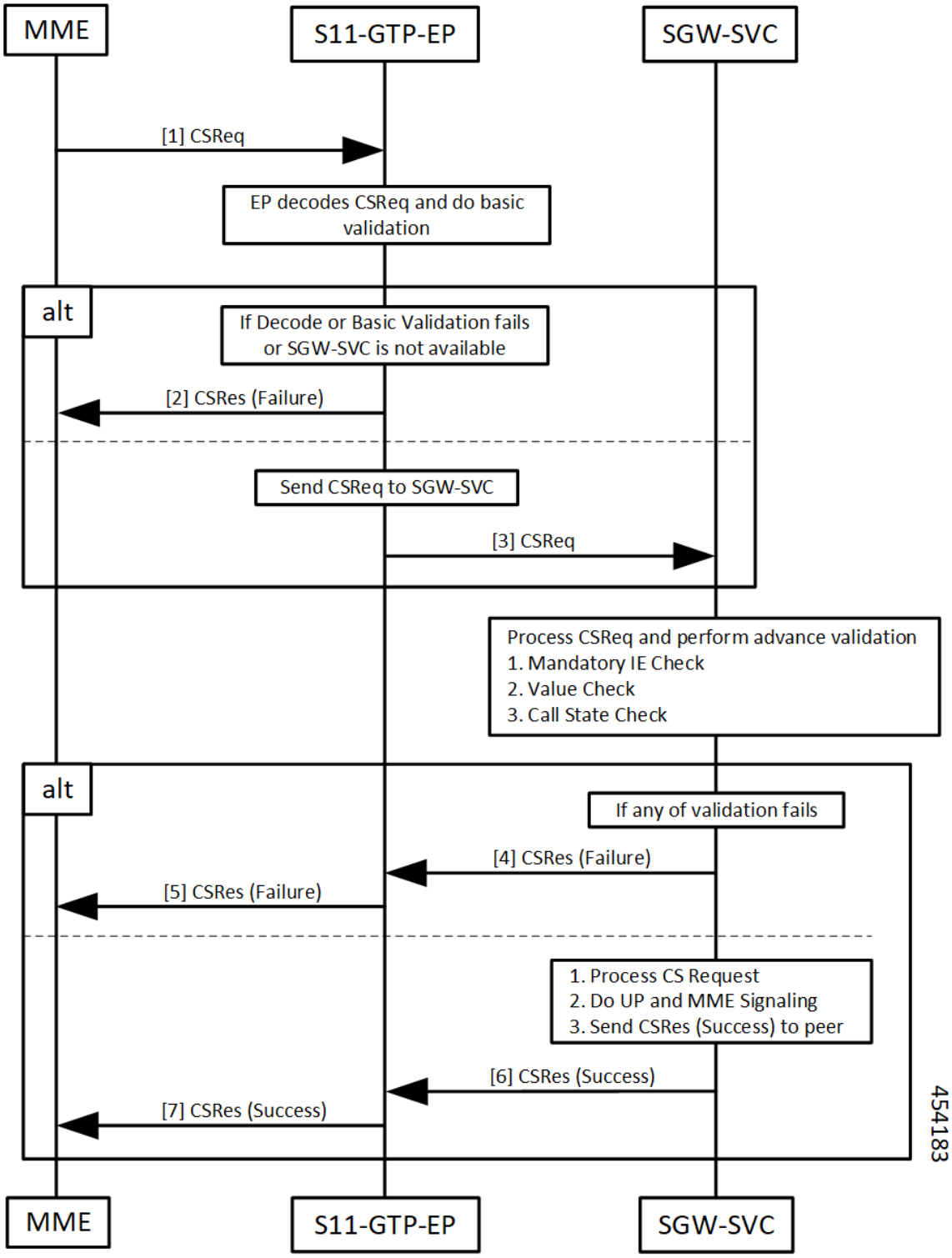


Table 118: Basic and Advance Validation - S11 Call Flow Description

Step	Description
1	The MME sends the Create Session Request to the S11-GTP-EP pod. The S11-GTP-EP pod decodes the Create Session Request and performs basic validation.
2	If decoding or basic validation fails, or if SGW-SVC is not available, the S11-GTP-EP pod sends Create Session Response failure message to the MME.
3	If Create Session Request basic validation is successful, the S11-GTP-EP pod forwards the Create Session Request to the SGW-SVC pod. SGW-SVC processes the Create Session Request and performs the following: • Mandatory IE check • Value check • Call State check
4, 5	If validation from Step 3 fails: • The SGW-SVC sends the Create Session Response failure message to the S11-GTP-EP pod. • The S11-GTP-EP pod forwards the Create Session Response failure message to the MME.
6, 7	If validation from Step 3 is successful, the SGW-SVC performs the following: • Processes Create Session Request • Performs UP and MME signaling • Sends Create Session Response success message to the S11-GTP-EP pod The S11-GTP-EP pod forwards the Create Session Response success message to the MME.

Basic and Advance Validation on SGW-Egress (S5) Call Flow

The following section describes the Basic and Advance Validation on SGW-Egress (S5) call flow.

Figure 57: Basic and Advance Validation - S5 Call Flow

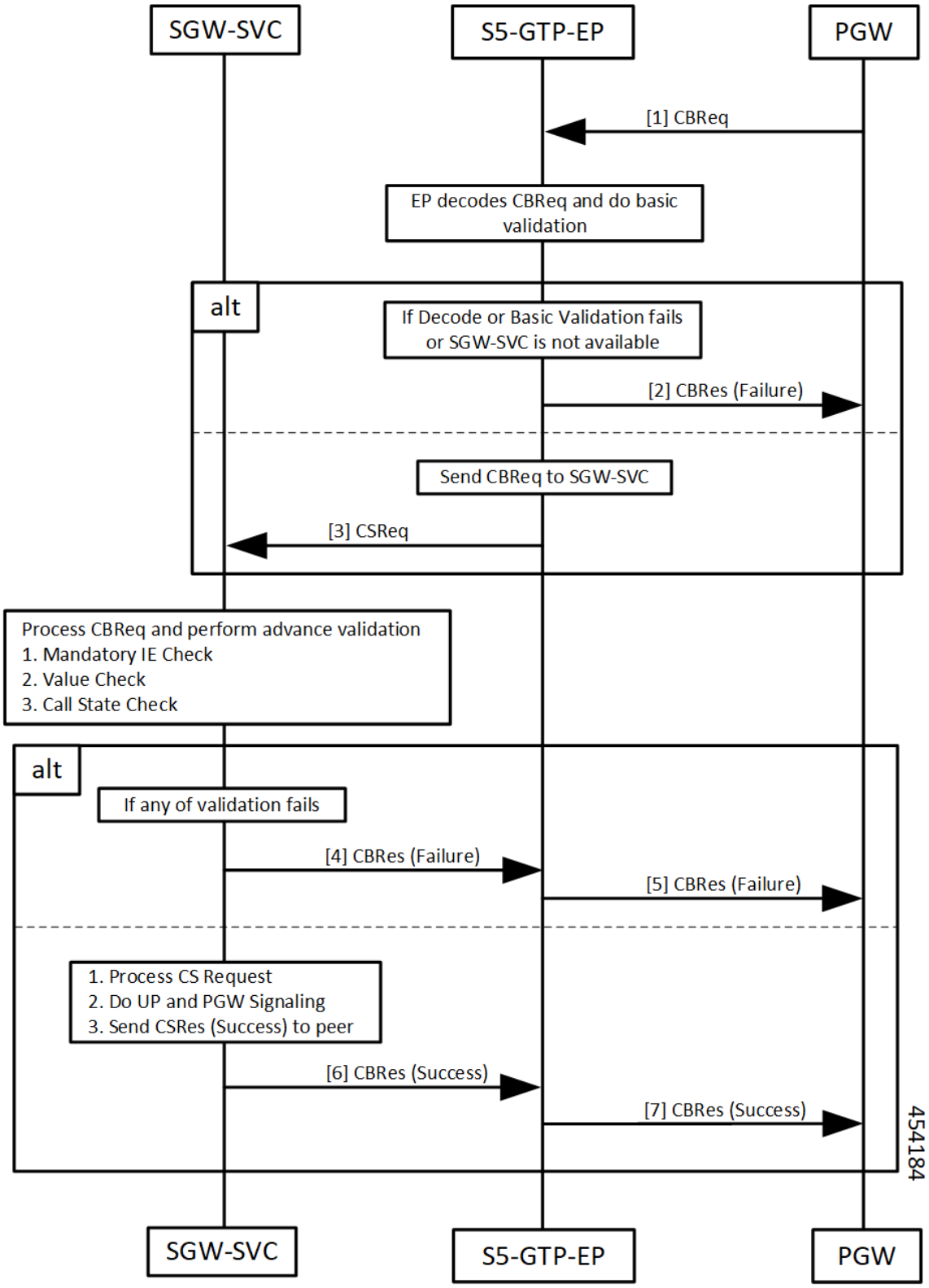
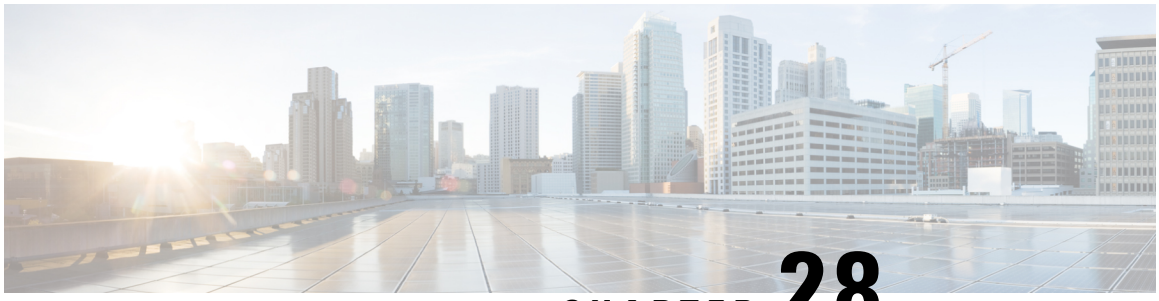


Table 119: Basic and Advance Validation - S5 Call Flow Description

Step	Description
1	The PGW sends the Create Bearer Request to the S5-GTPC-EP pod. The S5-GTPC-EP pod decodes the Create Bearer Request and performs basic validation.
2	If decoding or basic validation fails, or if SGW-SVC is not available, the S5-GTP-EP pod sends the Create Bearer Response failure message to the PGW.
3	If Create Bearer Request basic validation is successful, the S5-GTP-EP pod forwards the Create Bearer Request to the SGW-SVC pod. SGW-SVC processes the Create Bearer Request and performs the following: • Mandatory IE check • Value check • Call State check
4, 5	If validation from Step 3 fails: • The SGW-SVC sends the Create Bearer Response failure message to the S5-GTP-EP pod. • The S5-GTP-EP pod forwards the Create Bearer Response failure message to the PGW.
6, 7	If validation from Step 3 is successful, the SGW-SVC performs the following: • Processes Create Bearer Request • Performs UP and PGW signaling • Sends Create Bearer Response success message to the S5-GTP-EP pod The S5-GTP-EP pod forwards the Create Bearer Response success message to the PGW.



CHAPTER 28

IDFT Support

- Feature Summary and Revision History, on page 297
- Feature Description, on page 297
- How it Works, on page 298
- OAM Support, on page 308

Feature Summary and Revision History

Summary Data

Table 120: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 121: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

cnSGW-C supports Indirect Forwarding Tunnel (IDFT) Creation and Deletion for Pure-S call with dedicated bearers, with and without SGW relocation.

Standards Compliance

This feature complies with the following standards specifications:

- *3GPP TS 23.401 "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access"*
- *3GPP TS 23.402 "Architecture enhancements for non-3GPP accesses"*
- *3GPP TS 29.274 "Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C)"*
- *3GPP TS 23.214 "Architecture enhancements for control and user plane separation of EPC nodes"*
- *3GPP TS 29.244 "Interface between the Control Plane and the User Plane nodes"*
- *3GPP TS 24.008 "Mobile radio interface Layer 3 specification; Core network protocols; Stage 3"*

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

IDFT Support without SGW Relocation Call Flow

This section describes the IDFT Support without SGW Relocation call flow.

Figure 58: IDFT Support without SGW Relocation Call Flow

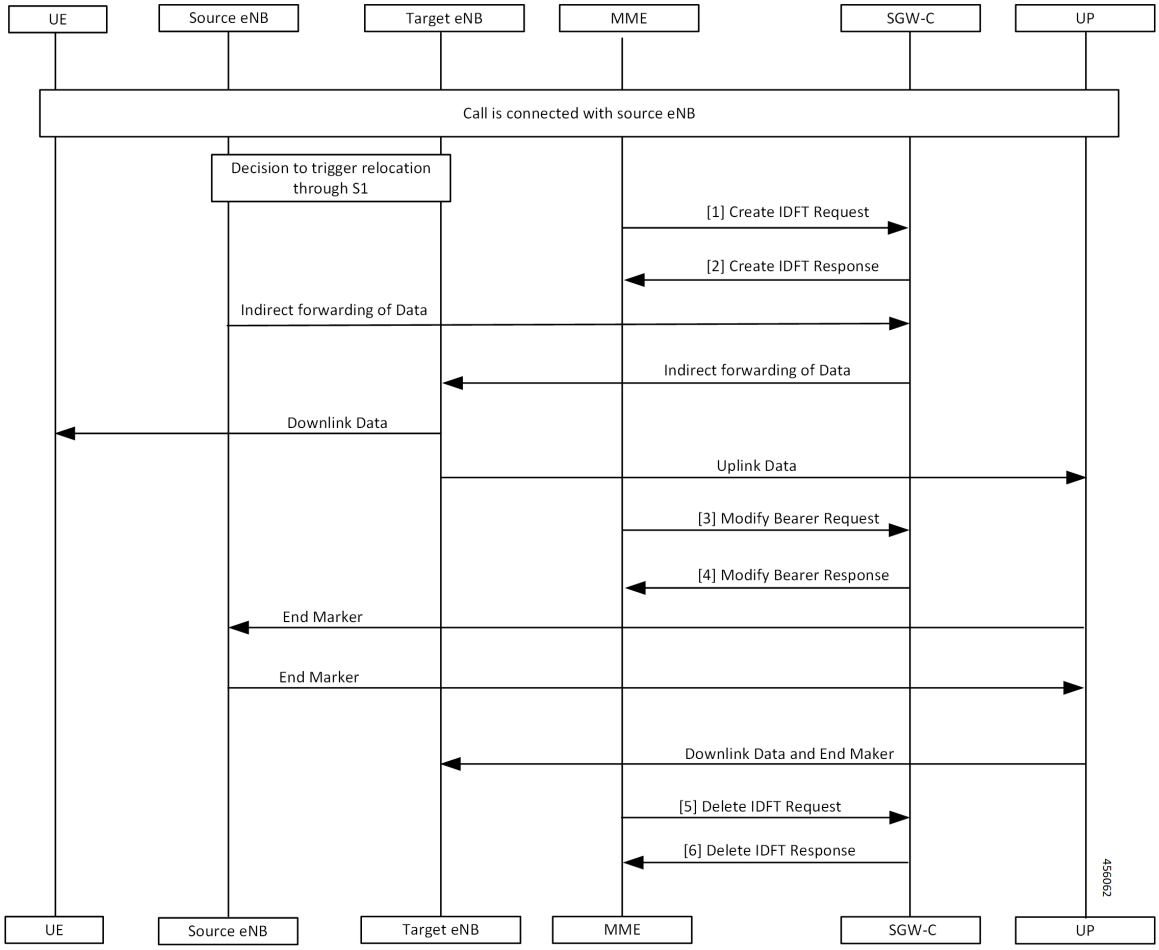


Table 122: IDFT Support without SGW Relocation Call Flow Description

Step	Description
1	Call is connected with the Source eNodeB and there's a decision to trigger relocation via S1. The MME sends the Create IDFT Request to the SGW-C.
2	The MME receives the Create IDFT Response from the SGW-C.
3	The indirect forwarding of the data starts from the Source eNodeB to the SGW-C. The indirect forwarding of the data starts from the SGW-C to the eNodeB. The Target eNodeB sends the Downlink Data to the UE. The Target eNodeB sends the Uplink Data to the UP. The MME sends the Modify Bearer Request to the SGW-C.

Step	Description
4	The SGW-C sends the Modify Bearer Response to the MME. The UP sends the End Marker to the Source eNodeB, and the Source eNodeB forwards the End Marker to the UP. The UP sends the Downlink Data and the End Marker to the Target eNodeB.
5	The MME sends the Delete IDFT Request to the SGW-C.
6	The MME receives the Delete IDFT Response from the SGW-C.

IDFT Support with SGW Relocation Call Flow

This section describes the IDFT Support with SGW Relocation call flow.

Figure 59: IDFT Support with SGW Relocation Call Flow

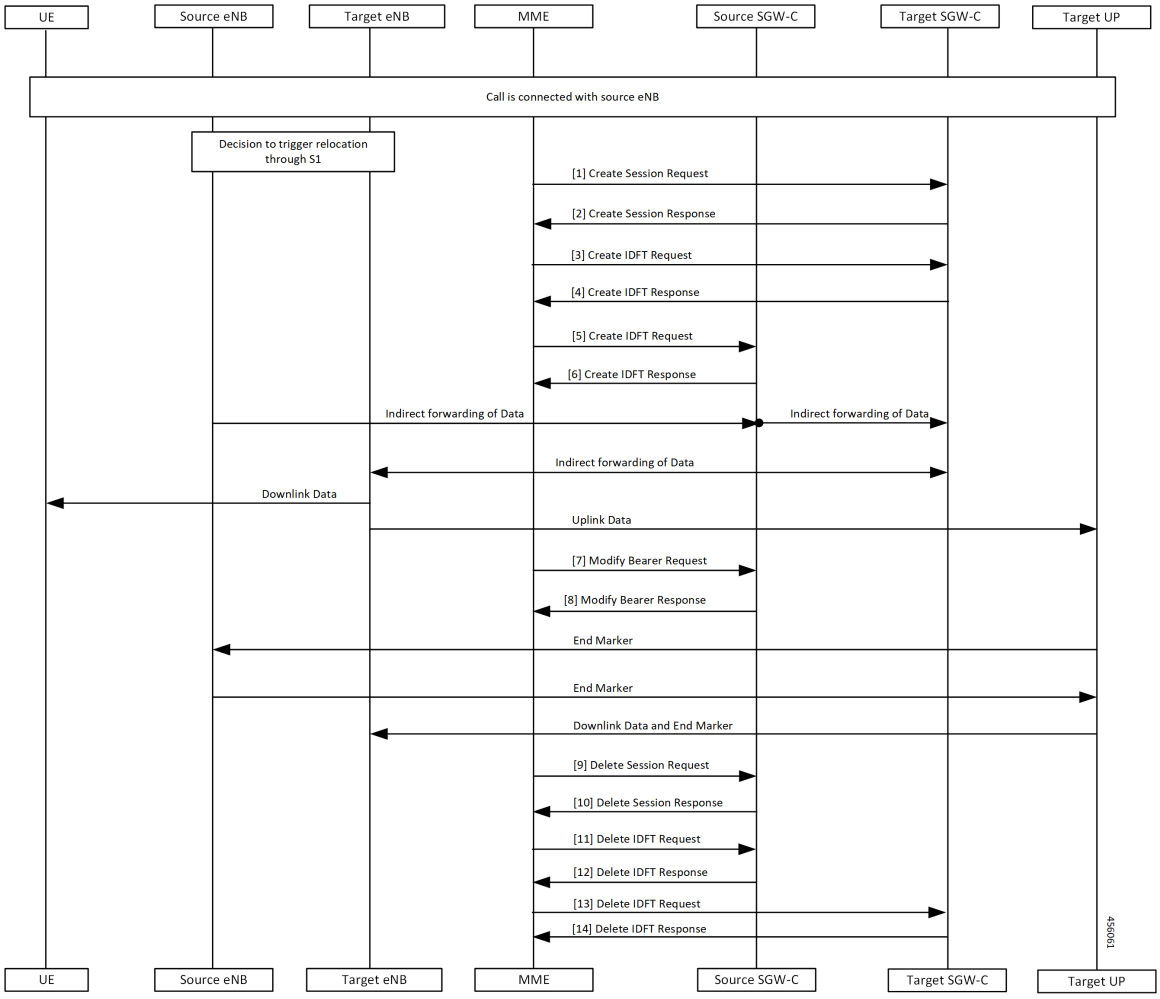


Table 123: IDFT Support with SGW Relocation Call Flow Description

Step	Description
1	Call is connected with the Source eNodeB and there's a decision to trigger relocation via S1. The MME sends the Create Session Request to the Target SGW-C.
2	The MME receives the Create Session Response from the Target SGW-C.
3	The MME sends the Create IDFT Request to the Target SGW-C.
4	The MME receives the Create IDFT Response from the Target SGW-C.
5	The MME sends the Create IDFT Request to the Source SGW-C.
6	The MME receives the Create IDFT Response from the Source SGW-C. The indirect forwarding of the data starts from the Source eNodeB to the Source SGW-C. The indirect forwarding of the data starts from the Source SGW-C to the Target SGW-C. The indirect forwarding of the data starts from the Target SGW-C to the Target eNodeB. The Target eNodeB sends the Downlink Data to the UE. The Target eNodeB sends the Uplink Data to the Target UP.
7	The MME sends the Modify Bearer Request to the Target SGW-C.
8	The Source SGW-C sends the Modify Bearer Response to the MME. The Target UP sends the End Marker to the Source eNodeB, and the Source eNodeB forwards the End Marker to the Target UP. The Target UP sends the Downlink Data and the End Marker to the Target eNodeB.
9	The MME sends the Delete Session Request to the Source SGW-C.
10	The MME receives the Delete Session Response from the Source SGW-C.
11	The MME sends the Delete IDFT Request to the Source SGW-C.
12	The MME receives the Delete IDFT Response from the Target SGW-C.
13	The MME sends the Delete IDFT Request to the Target SGW-C.
14	The MME receives the Delete IDFT Response from the Target SGW-C.

5G to 4G Handover Flow for Pure-S Call Flow

This section describes the 5G to 4G Handover flow for Pure-S call flow.

Figure 60: 5G to 4G Handover Flow for Pure-S Call Flow

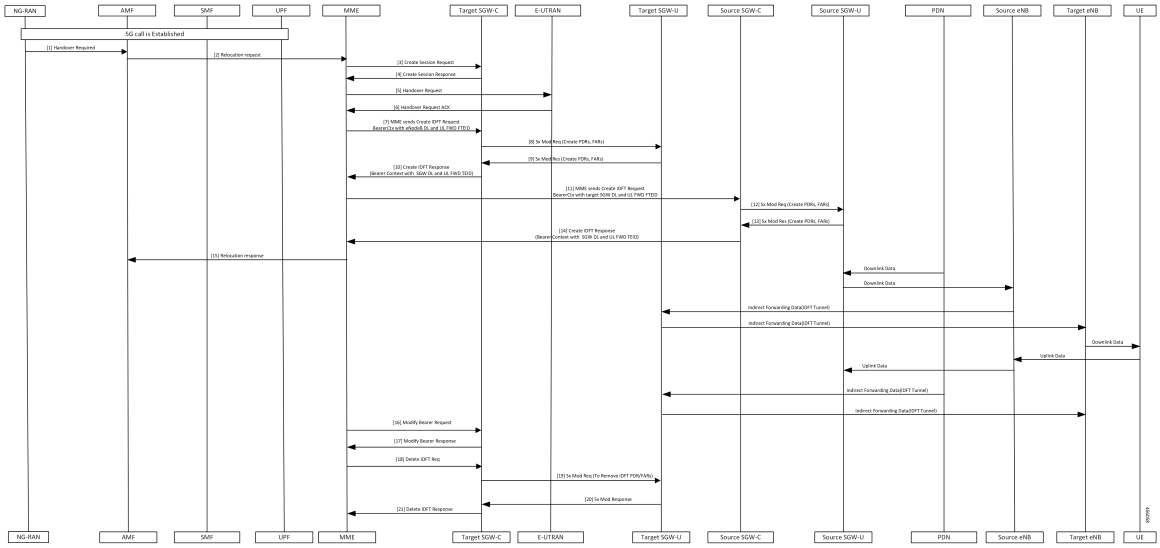


Table 124: 5G to 4G Handover Flow for Pure-S Call Flow Description

Step	Description
1	5G call is established. The NG-RAN sends the Handover Required message to the AMF.
2	The AMF sends the Relocation Request to the MME.
3	The MME sends the Create Session Request to the Target SGW-C.
4	The MME receives the Create Session Response from the Target SGW-C.
5	The MME sends the Handover Request to the E-UTRAN.
6	The MME receives the Handover Request ACK from the E-UTRAN.
7	The MME sends the Create IDFT Request with the Bearer Context with the eNodeB DL and UL FWD FTEID, to the Target SGW-C.
8	The Target SGW-C sends the Sx Modification Request with the Create PDRs and FARs to the Target SGW-U.
9	The Target SGW-U sends the Sx Modification Response with the Create PDRs and FARs to the Target SGW-C.
10	The Target SGW-C sends the Create IDFT Response with the Bearer Context with the SGW DL and UL FWD TEID, to the MME.
11	The MME sends Create IDFT Request with the Bearer Context, along with SGW DL and UL FWD FTEID, to the Source SGW-C.
12	The Source SGW-C sends the Sx Modification Request with Create PDRs and FARs, to the Source SGW-U.

Step	Description
13	The Source SGW-C receives the Sx Modification Response with Create PDRs and FARs, from the Source SGW-U.
14	The MME receives the Create IDFT Response with the Bearer Context, with the SGW DL and UL FWD TEID, from the Source SGW-C.
15	The MME sends the Relocation Response to the AMF. The PDN sends the Downlink Data to the Source SGW-U, and the Source SGW-U sends the Downlink Data to the Source eNodeB. The indirect forwarding data (IDFT Tunnel) starts from the Source eNodeB to the Target SGW-U, and from the Target SGW-U to the Target eNodeB. The Target eNodeB sends the Downlink Data to the UE. The UE sends the Uplink Data to the Source eNodeB, and the Source eNodeB sends the Uplink Data to the Source SGW-U. The PDN sends the indirect forwarding data to the Target SGW-U, and the Target SGW-U sends the indirect forwarding data to the Target eNodeB.
16	The MME sends the Modify Bearer Request to the Target SGW-C.
17	The MME receives the Modify Bearer Response from the Target SGW-C.
18	The MME sends the Delete IDFT Request to the Target SGW-C.
19	The Target SGW-C sends the Sx Modification Request with PDRs and FARs (to remove), to the Target SGW-U.
20	The Target SGW-C receives the Sx Modification Response from the Target SGW-U.
21	The MME receives the Delete IDFT Response from the Target SGW-C.

4G to 5G Handover Flow for Pure-S Call Flow

This section describes the 4G to 5G Handover flow for Pure-S call flow.

Figure 61: 4G to 5G Handover Flow for Pure-S Call Flow

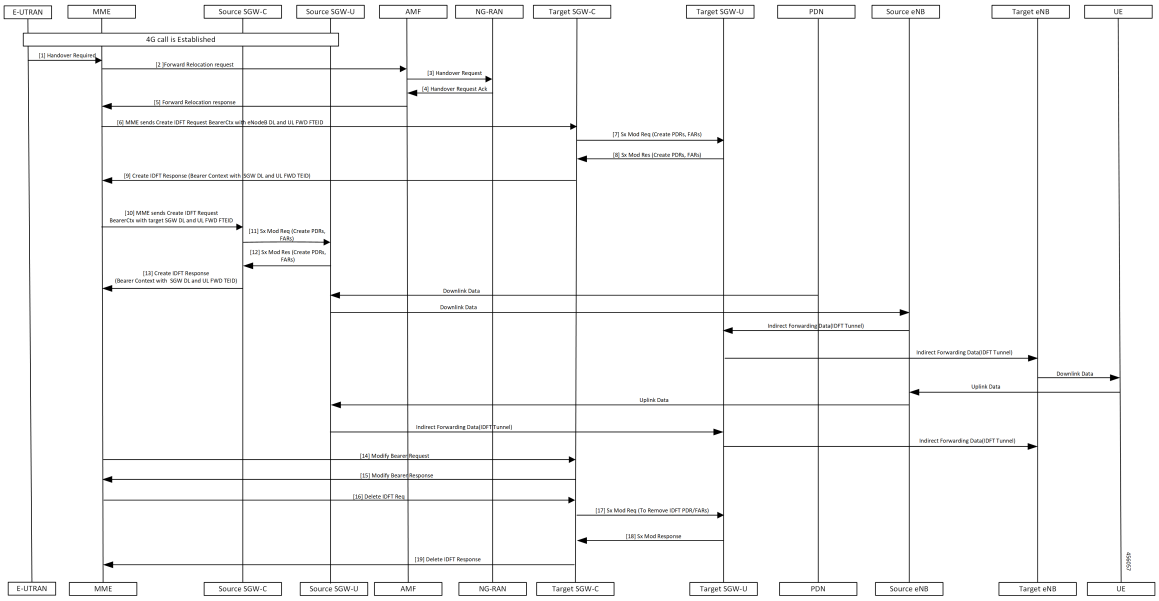


Table 125: 4G to 5G Handover Flow for Pure-S Call Flow Description

Step	Description
1	4G call is established. The E-UTRAN sends the Handover Required message to the MME.
2	The MME sends the Forward Relocation Request to the AMF.
3	The AMF sends the Handover Request message to the NG-RAN.
4	The AMF receives Handover Request ACK message.
5	The MME receives the Forward Relocation Response from the AMF.
6	The MME sends the Create IDFT Request with the Bearer Context with the eNodeB DL and UL FWD FTEID, to the Target SGW-C.
7	The Target SGW-C sends the Sx Modification Request with Create PDRs and FARs to the Target SGW-U.
8	The Target SGW-U sends the Sx Modification Response with Create PDRs and FARs to the Target SGW-C.
9	The Target SGW-C sends the Create IDFT Request with the Bearer Context along with eNodeB DL and UL FWD FTEID to the MME.
10	The MME sends the Create IDFT Request with the Bearer Context along with target SGW DL and UL FWD FTEID to the Source SGW-C.
11	The Source SGW-C sends the Sx Modification Request with Create PDRs and FARs to the Source SGW-U.

Step	Description
12	The Source SGW-C receives the Sx Modification Response with Create PDRs and FARs from the Source SGW-U.
13	The MME receives the Create IDFT Response with the Bearer Context along with SGW DL and UL FWD FTEID, from the Source SGW-C. The PDN sends the Downlink Data to the Source SGW-U, and the Source SGW-U sends the Downlink Data to the Source eNodeB. The indirect forwarding data (IDFT Tunnel) starts from the Source eNodeB to the Target SGW-U, and from the Target SGW-U to the Target eNodeB. The Target eNodeB sends the Downlink Data to the UE. The UE sends Uplink Data to the Source eNodeB, and the Source eNodeB sends the Uplink Data to the Source SGW-U. The Source SGW-U sends the indirect forwarding data to the Target SGW-U, and the Target SGW-U sends the indirect forwarding data to the Target eNodeB.
14	The MME sends the Modify Bearer Request to the Target SGW-C.
15	The MME receives the Modify Bearer Response from the Target SGW-C.
16	The MME sends the Delete IDFT Request to the Target SGW-C.
17	The Target SGW-C sends the Sx Modification Request to the Target SGW-U, to remove the PDRs and FARs.
18	The Target SGW-C receives the Sx Modification Response from the Target SGW-U.
19	The MME receives the Delete IDFT Response from the Target SGW-C.

Create IDFT (System-level) Call Flow

This section describes the Create IDFT (System-level) call flow.

Figure 62: Create IDFT (System-level) Call Flow

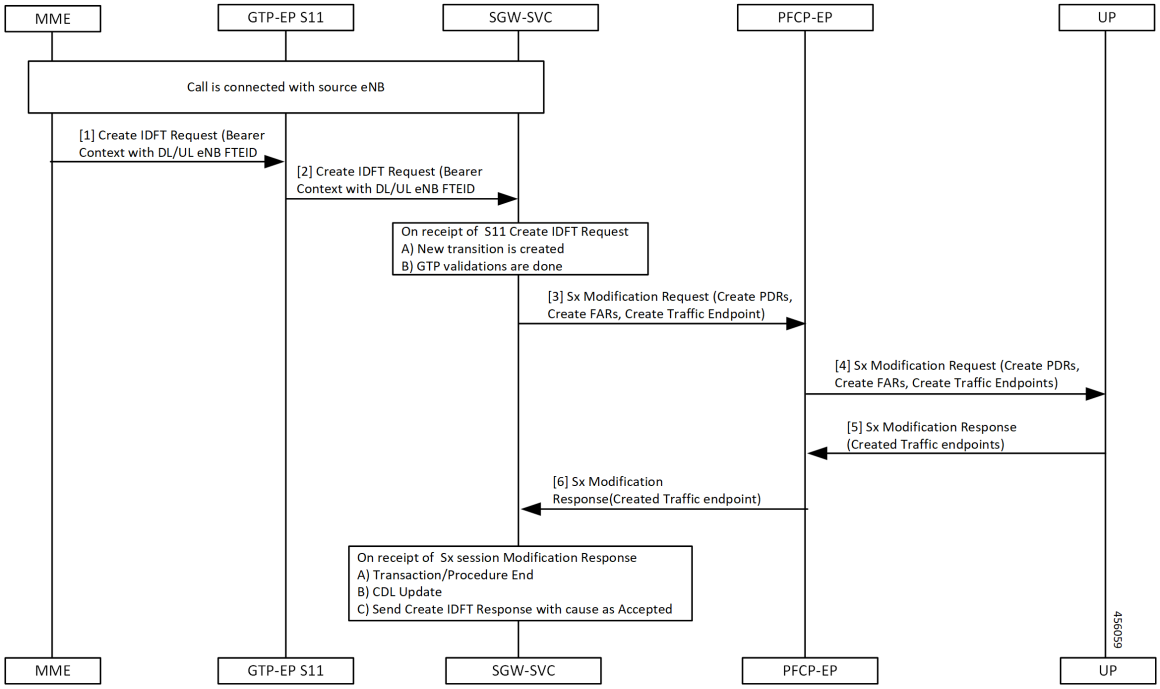


Table 126: Create IDFT (System-level) Call Flow Description

Step	Description
1	Call is connected with the Source eNodeB. The MME sends the S11 Create IDFT Request with the Bearer Context with a DL/UL enB FTEID, to the GTP-EP S11.
2	The GTP-EP S11 sends the S11 Create IDFT Request with the Bearer Context with a DL/UL enB FTEID, to the SGW-SVC. The SGW-SVC receives the S11 Create IDFT Request and performs the following: • Creates a new transaction • Completes GTP validations
3	The SGW-SVC sends the Sx Modification Request with Create PDRs, Create FARs, and Create Traffic Endpoints, to the PFCP-EP.
4	The PFCP-EP sends the Sx Modification Request with Create PDRs, Create FARs, and Create Traffic Endpoints, to the UP.
5	The UPF sends the Sx Session Modification Response with Created Traffic endpoints, to the PFCP-EP.

Step	Description
6	The SGW-SVC receives the Sx Session Modification Response from the PFCP-EP and performs the following: • Ends the transaction/procedure • Updates the CDL • Sends the Create IDFT Response with cause as Accepted

Delete IDFT (System-level) Call Flow

This section describes the Delete IDFT (system-level) call flow.

Figure 63: Delete IDFT (System-level) Call Flow

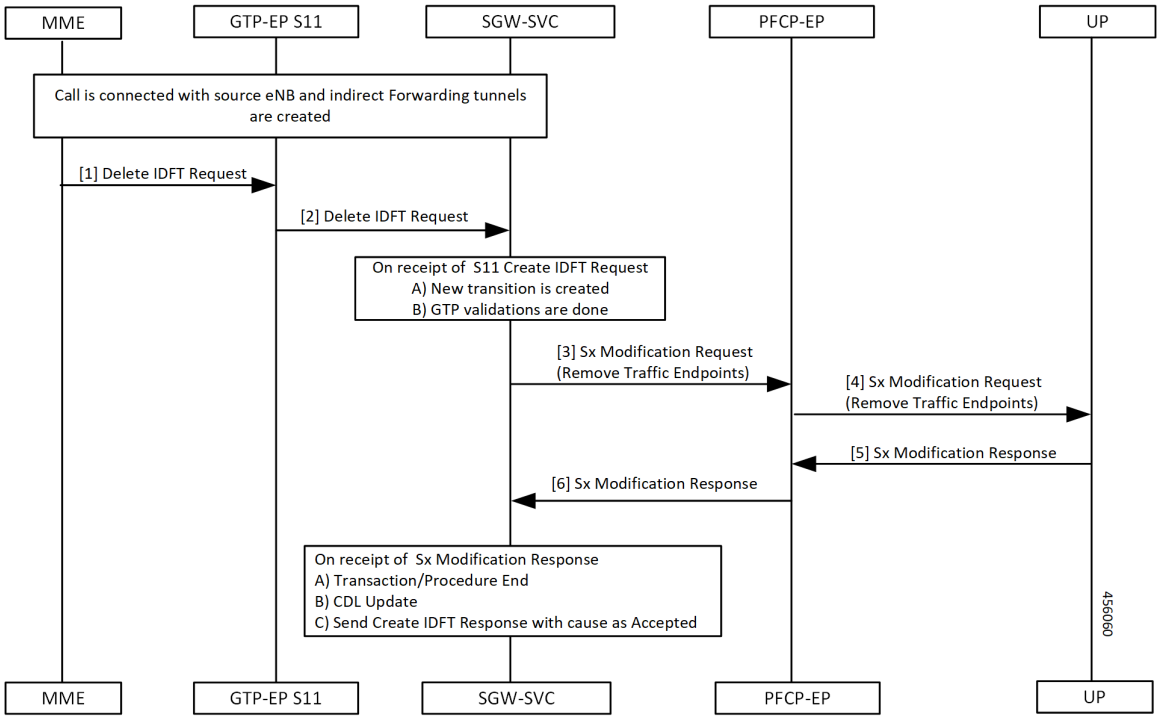


Table 127: Delete IDFT (System Level Flow) Call Flow Description

Step	Description
1	Call is connected with the Source eNodeB and indirect forwarding tunnels are created. MME sends the S11 Delete IDFT Request with Bearer Context with a DL/UL enB FTEID, to the GTP-EP S11.

Step	Description
2	The GTP-EP S11 forwards the S11 Delete IDFT Request with Bearer Context with a DL/UL enB FTEID, to the SGW-SVC. The SGW-SVC receives the S11 Delete IDFT Request and performs the following: • Creates a new transaction • Completes the GTP validations
3	The SGW-SVC sends the Sx Modification Request with Remove Traffic Endpoints, to the PFCP-EP.
4	The PFCP-EP sends the Sx Modification Request with Remove Traffic Endpoints, to the UP.
5	The UP sends the Sx Session Modification Response to the PFCP-EP.
6	The SGW-SVC receives the Sx Session Modification Response from the PFCP-EP and performs the following: • Ends the transaction/procedure • Updates the CDL • Sends the Delete IDFT Response with cause as Accepted

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Viewing IDFT Configuration

This section describes the command to view IDFT configurations.

Non- active IDFT for the UE

Command: **show subscriber namespace sgw imsi 123456789012345**

Output:

```
{ "subResponses": [ { "status": true,
…
…
"pdnInfoList": { "totalPdn": 1,
"bearerInfoList": {
"totalBearer": 1, "bearerInfo": [ { "bearerId": "Bearer-1", "state": "Connected",
…
…
} }
]
}
```

Active IDFT for the UE with one PDN having one bearer

Command: **show subscriber namespace sgw imsi 123456789012345**

Output:

```
{ "subResponses": [ { "status": true,
…
…
"pdnInfoList": { "totalPdn": 1,
"bearerInfoList": {
"totalBearer": 1, "bearerInfo": [ { "bearerId": "Bearer-1", "state": "Connected",
…
…
"IndirectForwardingInfo": {
"UplinkInfo":{
"localTeid": "[0x1100000e] 285212686", "localIPv4Address": "209.165.201.8", "remoteTeid":
"[0x1100000f] 285212687", "remoteIPv4Address": "209.165.201.8",
}
"DownlinkInfo":{
"localTeid": "[0x1100000e] 285212686", "localIPv4Address": "209.165.201.8", "remoteTeid":
"[0x1100000f] 285212687", "remoteIPv4Address": "209.165.201.8",
}
}
}
]
}
```

Active IDFT for the UE with one PDN having one bearer in downlink direction

Command: **show subscriber namespace sgw imsi 123456789012345**

Output:

```
{ "subResponses": [ { "status": true,
…
…
"pdnInfoList": { "totalPdn": 1,
"bearerInfoList": {
"totalBearer": 1, "bearerInfo": [ { "bearerId": "Bearer-1", "state": "Connected",
…
…
"IndirectForwardingInfo": {
"DownlinkInfo":{
"localTeid": "[0x1100000e] 285212686",
"localIPv4Address": "209.165.201.8",
"remoteTeid": "[0x1100000f] 285212687",
"remoteIPv4Address": "209.165.201.8",
}
}
}
]
}
```

Active IDFT for one bearer for the UE with one PDN having two bearers

Command: **show subscriber namespace sgw imsi 123456789012345**

Output:

```
"subResponses": [ { "status": true,
…
…
"pdnInfoList": { "totalPdn": 1,
"bearerInfoList": {
"totalBearer": 2, "bearerInfo": [ { "bearerId": "Bearer-1", "state": "Connected",
…
…
```

```
"IndirectForwardingInfo": {
“UplinkInfo”:{
"localTeid": "[0x1100000e] 285212686", "localIPv4Address": "209.165.201.8", "remoteTeid":
"[0x1100000f] 285212687", "remoteIPv4Address": "209.165.201.8",
}
“DownlinkInfo”:{
"localTeid": "[0x1100000e] 285212686", "localIPv4Address": "209.165.201.8", "remoteTeid":
"[0x1100000f] 285212687", "remoteIPv4Address": "209.165.201.8",
}
}
"bearerInfo": [ { "bearerId": "Bearer-2", "state": "Connected",
…
}
}
]
}
```

Note The displayed IndirectForwardingInfo block is only for bearers having indirect forwarding tunnels.

Failure Handling

cnSGW-C supports failure handling for creating or deleting IDFT request procedure.

Following are the failure types that can occur during message processing:

- Advance validation failure on request and response
- Retransmissions timeout
- Transaction SLA
- Failure reported from peer (UP/PGW/MME), depending on the stage of message processing.

The following table depicts the behavior of cnSGW-C during different failure scenarios in call processing.

Failure Scenario	SGW-SVC behavior	Signaling (S11)
1. Create IDFT Request advance validation failure. 2. cnSGW doesn’t have a bearer context for any of the EBIs received in Create IDFT.	Sends failure or No signaling over Sx.	Negative Create IDFT response.

Failure Scenario	SGW-SVC behavior	Signaling (S11)
Single PDN **1.** Sx Session Modify Request (for example, IPC, Retransmission, Internal Failure) with single PDN **2.** Sx Session Modify Response (Cause!= ACCEPTED) with single PDN **3.** Sx Modification Response validation failure	Sends failure. Sends Context not found of nonexisting EBI. Clear the PDN if sxCause = Context Not Found.	Negative Create IDFT response. DBR and DSR over S11 and S5 when *sxCause = Context Not Found.*
Multi PDN (Partial Failure) **1.** Partial Existing PDN: Continue with existing PDN **2.** Sx Session Modify Request (for example, IPC, Retransmission Timeout, Internal Failure) for some PDNs **3.** Sx Session Modify Response (Cause!= ACCEPTED) for some PDN **4.** Sx Modification Response validation failure	Send Context Not Found for nonexisting PDNs. Send failure in Bearer Context for PDNs for which Sx Modification Request fails.	Partially Accepted Create IDFT Response. DBR and DSR over S11 and S5 for the PDN for which *sxCause = Context Not Found.*
Multi PDN (Complete Failure): **1.** Partial Existing PDN: Continue with existing PDN. **2.** Sx Session Modify Request (for example, IPC, Retransmission Timeout, Internal Failure) **3.** Sx Session Modify Response (Cause!= ACCEPTED) **4.** Sx Modification Response validation failure	Send Context Not Found for nonexisting PDNs. Send failure in Bearer Context for PDNs which has Sx Modification Request fails.	Negative Create IDFT Response. DBR and DSR over S11 and S5 for the PDN for which *sxCause = Context Not Found.*
Delete IDFT Request Advance validation failure.	Send failure or No signaling over Sx.	Negative Delete IDFT response.

Failure Scenario	SGW-SVC behavior	Signaling (S11)
1. Single and Multi-PDN 2. Sx Session Modify Request (for example, IPC, Retransmission Timeout, Internal Failure) 3. Sx Session Modify Response (Cause!= ACCEPTED)	Ignore Failure.	Positive Delete IDFT Response. DBR and DSR over S11 and S5 for the PDN for which *sxCause = Context Not Found.*

Bulk Statistics Support

The following statistics are supported for the IDFT Support feature.

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="create_indirect_data_forwarding_tunnel",status="attempted",sub_fail_reason=""}
 3
```

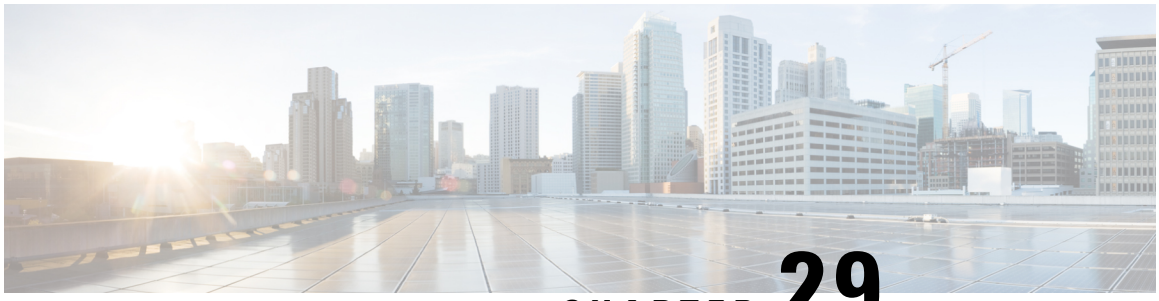
```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",instance_id="0",
interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"create_indirect_data_forwarding_tunnel",status="success",sub_fail_reason=""} 2
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",instance_id="0",
interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"delete_indirect_data_forwarding_tunnel",status="attempted",sub_fail_reason=""}
1
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="delete_indirect_data_forwarding_tunnel",status="success",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",instance_id="0",
interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"indirect_data_forwarding_tunnel_guard_timer_expiry",status="attempted",sub_fail_reason=""}
 1
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",instance_id="0",
interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",sgw_procedure_type=
"indirect_data_forwarding_tunnel_guard_timer_expiry",status="success",sub_fail_reason=""}
 1
```



CHAPTER 29

Idle Session Timeout Settings

- Feature Summary and Revision History, on page 313
- Feature Description, on page 313
- How it Works, on page 314
- Feature Configuration, on page 320

Feature Summary and Revision History

Summary Data

Table 128: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 129: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

The stale session timeout determines the duration for which the SGW-U sessions can remain inactive before they are terminated. On the cnSGW-C platform, the SubscribeCtx represents the subscriber session. The SGW-U establishes a connection with peers, such as:

- MME and PGW using the S11 or S4 interface
- PGW on the S5 or S8 interface

When the peers delete the peer session, the SGW-U doesn't receive the deletion message or inadvertently misses them. In such situations, the SGW-U sessions remain idle and continue to receive the calls but do not process or respond to the request. To prevent the stale sessions from using the resources, the idle timeout feature enables the SGW-U to receive new subscriber session requests after deleting the old or stale sessions.

How it Works

This section describes how this feature works.

A subscriber session is idle when data traffic activity is not steered towards it as it is inactive for a stipulated time.

The session manager on the user plane tracks the state of the call line. Sessions for which the session manager does not record the call line data traffic are determined as idle. Using the idle session timeout configuration, you can set the time interval for which the session can remain idle before it times out. The idle timeout configuration is set when the session is established. The SGW-U sends the timeout configuration to the user plane in the Sx Session Establishment Request. In case of multi-PDN calls, the calls directed towards a stale session are cleared after the inactivity report is generated for all PDNs.

Every second, the SGW-U monitors the data traffic activity to determine the session's idleness status. On identifying a stale session, the user plane updates the User Plane Inactivity Report in the Sx Session Usage Report and sends it to cnSGW-C to convey that the session is idle. Further, the cnSGW-C initiates a session deletion request towards its peers.

Based on the network environment, configure the idle timeout configuration in seconds. The accepted range of the timeout value is 1–4294967295 seconds. The timeout configuration is applicable at the SGW-U service profile level enabling the idle timeout handling for the set of subscribers handled by the SGW-U service.

Call Flows

This section describes the key call flows for this feature.

Inactivity Report Call Flow

This section describes the Inactivity Report call flow.

Figure 64: Inactivity Report Call Flow



Table 130: Inactivity Report Call Flow Description

Step	Description
1	The MME sends the Create Session Request to the GTP-EP (S11).
2	The GTP-EP (S11) forwards the Create Session Request to the SGW.
3	The SGW-SVC sends the Session Idle Timer in IE = USER INACTIVITY as part of Sx Session Establishment Request to the Protocol.
4	The SGW-U reads the Session Idle Timer from USER INACTIVITY and stores it at the CLP level. The UPF sends the Sx Session Report Request to the Protocol.
5	The Protocol sends the Sx Session Report Request to the SGW-SVC.
6	The SGW-SVC sends the Sx Session Report Response to the Protocol.
7	The Protocol sends the Sx Session Report Response to the UPF.

Step	Description
8	The SGW-SVC sends the Delete Session Request to the GTPC-EP.
9	The GTPC-EP sends the Delete Session Request to the PGW.
10	The SGW-SVC sends the Delete Bearer Request to the GTP-EP.
11	The GTP-EP sends the Delete Bearer Request to the MME.
12	The MME sends the Delete Bearer Response to the GTP-EP.
13	The GTP-EP sends the Delete Bearer Response to the SGW-SVC.
14	The PGW sends the Delete Session Response to the GTPC-EP.
15	The GTPC-EP sends the EGTP Delete Session Response to the SGW-SVC.
16	The SGW-SVC sends the Sx Session Deletion Request to the Protocol.
17	The Protocol sends the Sx Session Deletion Request to the UPF.
18	The UPF sends the Sx Session Deletion Response to the Protocol.
19	The Protocol sends the Sx Session Deletion Response to the SGW-SVC.

Idle Timer Handling on UPF Call Flow

This section describes the call flow when the idle timer is received in the Create Session Request on the UPF.

Figure 65: Idle Timer Handling on UPF Call Flow

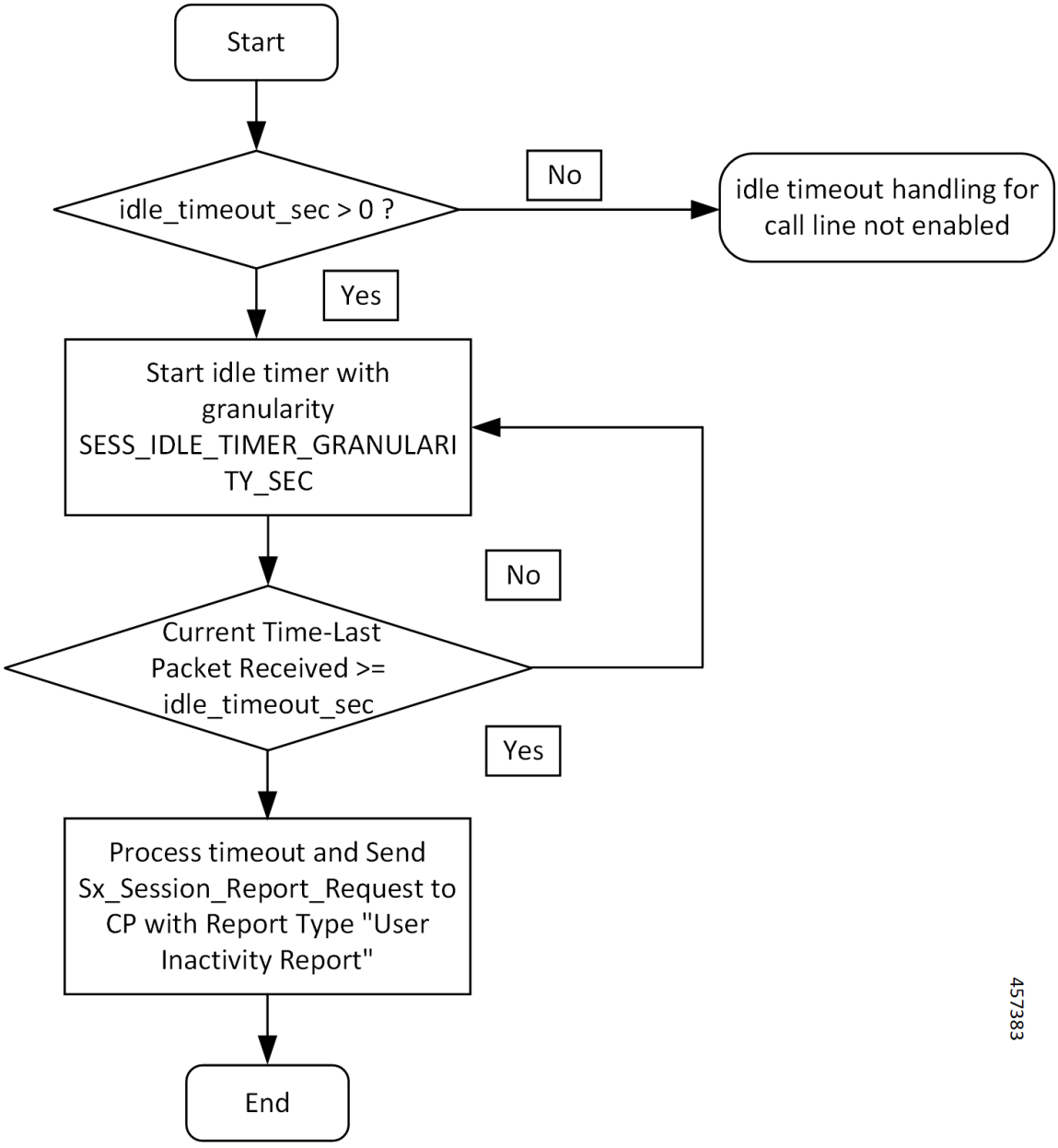


Table 131: Idle Timer Handling on UPF Call Flow Description

Step	Description
1	If the value of idle_timeout_sec is greater than zero, the timer is started on UPF with granularity of one second. Else, idle timeout is disabled.

Step	Description
2	The timer timeouts every second. Checks if the difference in current time and last packet received is greater than idle_timeout_sec or not. If the time difference is greater than idle_timeout_sec, then UP sends the Sx Session Report Request with Report Type = UPIR (Inactivity Report) to CP (cnSGW-C).

Reactivity Report Call Flow

This section describes the Reactivity Report call flow.

Figure 66: Reactivity Report Call Flow

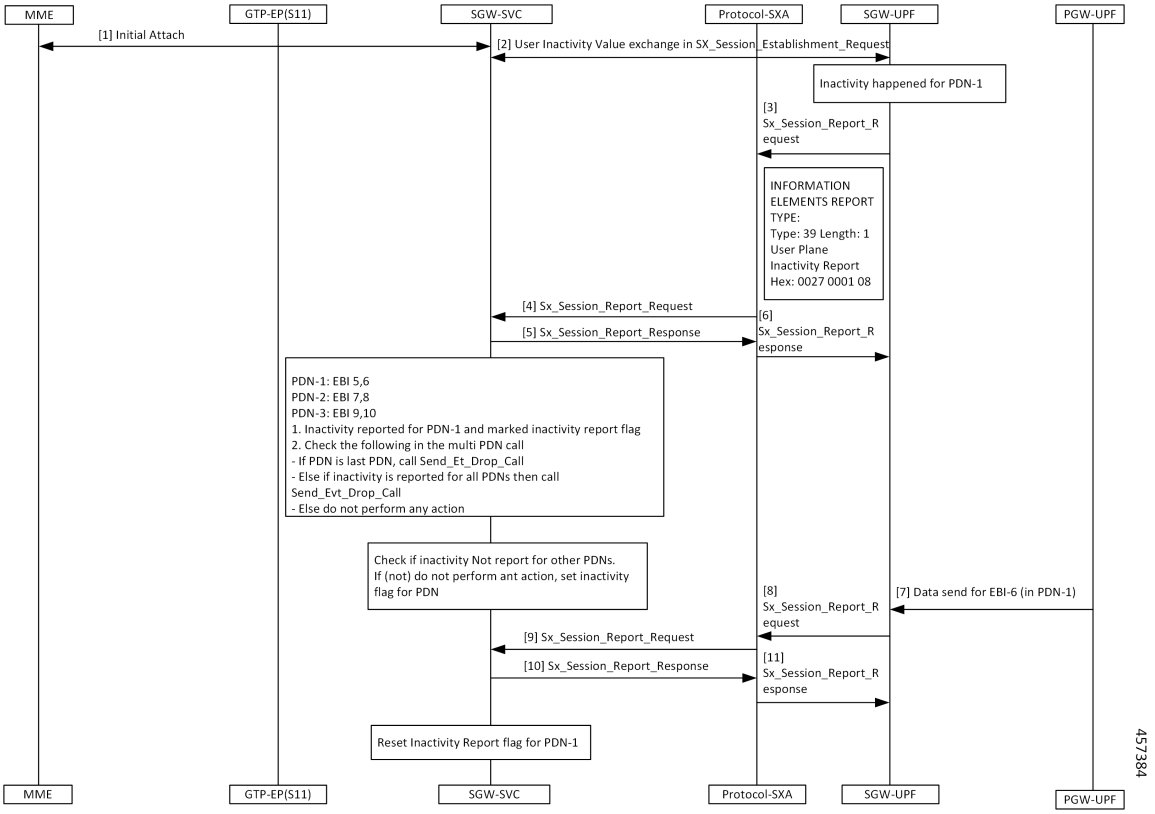


Table 132: Reactivity Report Call Flow Description

Step	Description
1	The MME and the SGW-SVC process the initial attach request.
2	The Protocol-SXA and the SGW-UPF perform the user inactivity exchange in the Sx Session Establishment Request.
3	If the inactivity is observed in PDN-1, the SGW-UPF sends the Sx_Session_Report_Request with type User Plane Inactivity Report to the Protocol-SXA.

Step	Description
4	The Protocol-SXA sends the Sx_Session_Report_Request to the SGW-SVC.
5	The SGW-SVC sends the Sx_Session_Report_Response to the Protocol-SXA.
6	The Protocol-SXA forwards the Sx_Session_Report_Response to the SGW-UPF.
7	The PGW-UPF sends the data for the EBI-6 (in PDN-1) to the SGW-UPF.
8	The SGW-UPF sends the Sx_Session_Report_Request with IE Report-Type = User Plane Re-Activity Report to the Protocol-SXA.
9	The Protocol-SXA sends the Sx_Session_Report_Request with IE Report-Type = User Plane Re-Activity Report to the SGW-SVC.
10	The SGW-SVC responds with the Sx_Session_Report_Response to the Protocol-SXA.
11	The Protocol-SXA sends the Sx_Session_Report_Response to the SGW-UPF. After receiving the Sx_Session_Report_Response on the control plane, SGW-SVC clears the Inactivity Report Flag for the PDN.

Clear Call Handling Call Flow

This section describes the Clear Call Handling call flow.

Figure 67: Clear Call Handling Call Flow

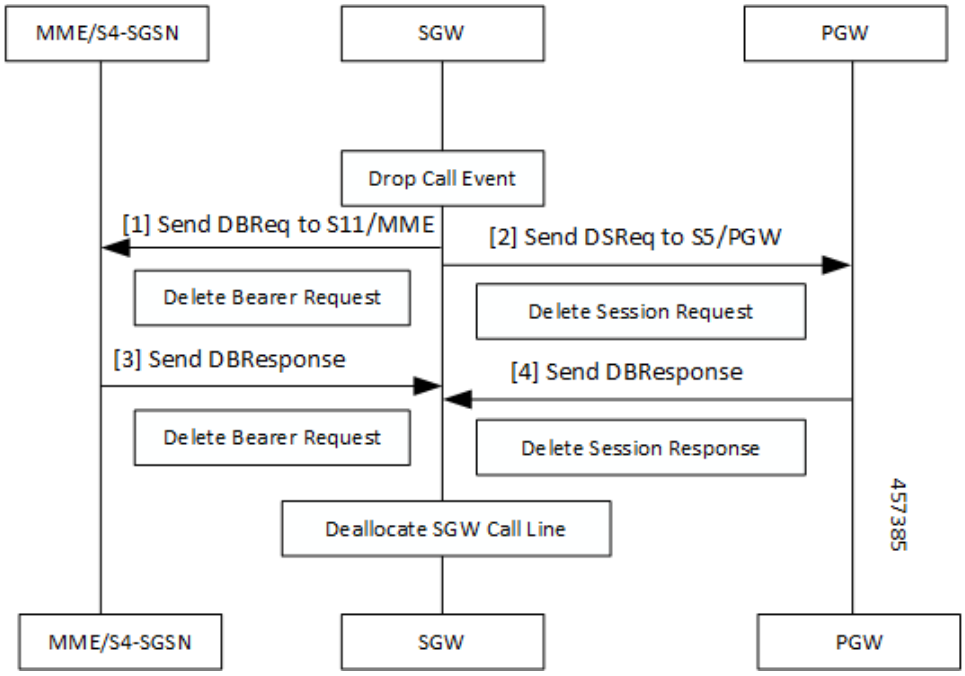


Table 133: Clear Call Handling Call Flow Description

Step	Description
1	On the SGW, the control plane receives the User Plane Inactivity Report in Sx Session Report Request. The control plane evaluates if PDN in the request is the latest PDN, or the inactivity report has already reported other PDNs. The control plane initiates the ClearCall procedure.
2	After receiving the ClearCall message, the cnSGW-C triggers a Session Deletion Request to its peers.
3	The SGW sends the Delete Bearer Request to the S11/MME.
4	The SGW sends the Delete Session Request to the S5/PGW.
5	On receiving the Delete Session Request, SGW clears resources on the UPF by sending a Sx Session Delete Request.

Feature Configuration

To configure this feature, use the following configuration:

```
config
  profile sgw sgw_group_name
    session-idle-timer session_idle_timer
    end
```

NOTES:

- **session-idle-timer** *session_idle_timer*—Specify the maximum duration in seconds for which a session remains idle. After the configured time is reached, the system automatically terminates the session. The accepted range contains integers in the range of 1–4294967295. The default value is zero indicating that the idle session is disabled.

Configuration Example

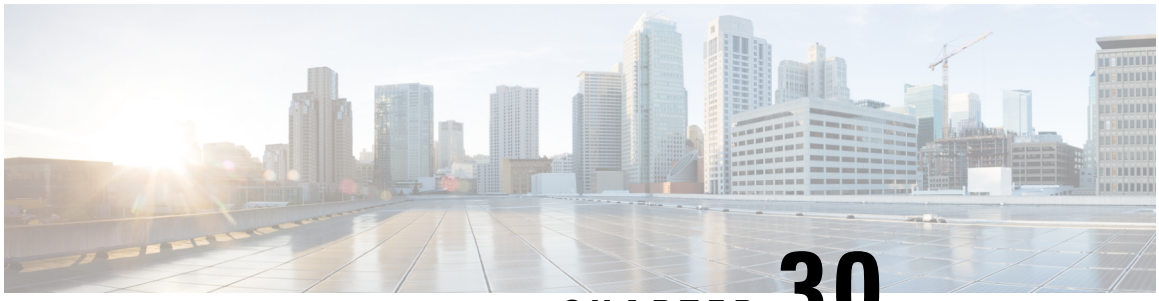
The following is an example configuration.

```
config
   profile sgw sgw1
     session-idle-timer 1000
     end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw sgw1 session-idle-timer
profile sgw sgw1
session-idle-timer 1000
```



CHAPTER 30

Initial Attach Support

- Feature Summary and Revision History, on page 321
- Feature Description, on page 322
- How it Works, on page 322
- Support for Backoff Timer, Origination TimeStamp, and MaxWait Time, on page 325

Feature Summary and Revision History

Summary Data

Table 134: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 135: Revision History

Revision Details	Release
Added support for the Information Elements (IE): • Backoff Timer • Origination Time Stamp • Maximum Wait Time	2021.02.3
First introduced.	2020.04

Feature Description

cnSGW-C supports handling of Initial Attach Create Session Request. As a part of this feature, cnSGW-C supports receiving Create Session Request from the MME through the EGTP endpoint. Further, cnSGW-C decodes the UDP message and converts the message into gRPC message for internal message processing.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flow for this feature.

Initial Attach Call Flow

This section describes the Initial Attach call flow.

Figure 68: Initial Attach Call Flow

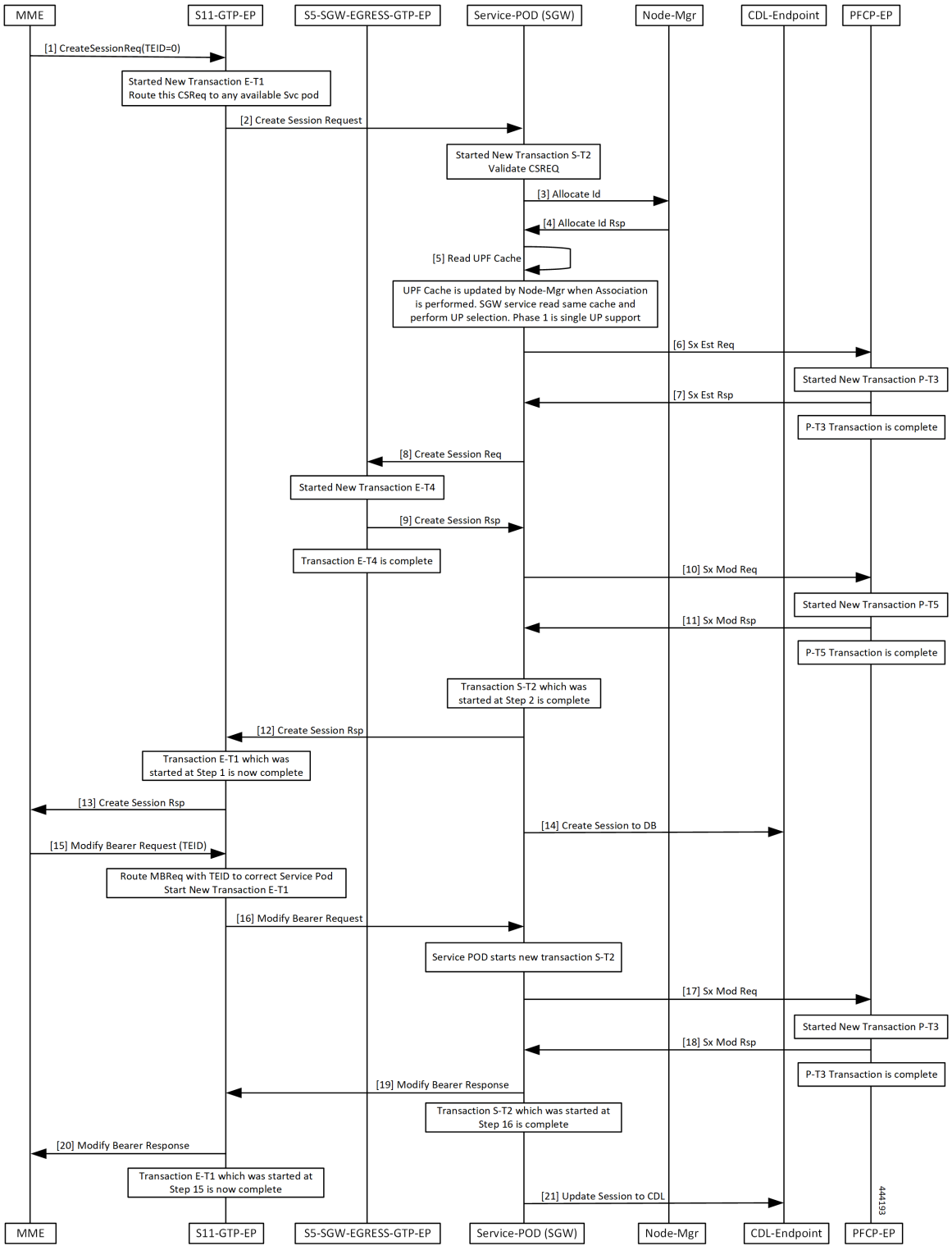


Table 136: Initial Attach Call flow Description

Step	Description
1	The MME sends the Create Session Request with TEID value zero to the S11-GTP-EP.
2	Transaction E-T1 is started. The S11-GTP-EP forwards the Create Session Request to the Service-POD (SGW).
3	Transaction S-T2 is started to validate the Create Session Request. The Service-POD (SGW) sends the Allocate Id Request to the Node-Mgr.
4	The Node-Mgr sends the Allocate Id Response to the Service-POD (SGW).
5	The Service-POD (SGW) reads the cache to perform the UPF selection.
6	The Service-POD (SGW) sends the Sx Establishment Request to the PFCP-EP.
7	Transaction P-T3 is started. The PFCP-EP sends the Sx Establishment Response to the Service-POD (SGW).
8	Transaction P-T3 is completed. The Service-POD (SGW) sends the Create Session Request to the S5-SGW-EGRESS-GTP-EP.
9	Transaction E-T4 is started. The S5-SGW-EGRESS-GTP-EP sends the Create Session Response to the Service-POD (SGW).
10	Transaction E-T4 is completed. The Service-POD (SGW) sends the Sx Modification Request to the PFCP-EP.
11	Transaction P-T5 is started. The PFCP-EP sends the Sx Modification Response to the Service-POD (SGW).
12	Transactions P-T5 and S-T2 are completed. The Service-POD (SGW) sends the Create Session Response to the S11-GPT-EP.
13	Transaction E-T1 is completed. The S11-GPT-EP forwards the Create Session Response to the MME.
14	The Service-POD (SGW) sends the Create Session to DB message to the CDL-Endpoint.
15	The MME sends the Modify Bearer Request with TEID to the S11-GTP-EP.
16	Transaction E-T1 is started. The S11-GTP-EP sends the Modify Bearer Request to the Service-POD (SGW).
17	Transaction S-T2 is started. The Service-POD (SGW) sends the Sx Modification Request to the PFCP-EP.

Step	Description
18	Transaction P-T3 is started. The PFCP-EP sends the Sx Modification Response to the Service-POD (SGW).
19	Transaction P-T3 is completed. The Service-POD (SGW) sends the Modify Bearer Response to the S11-GTP-EP.
20	Transaction S-T2 is completed. The S11-GPT-EP forwards the Modify Bearer Response to the MME.
21	Transaction E-T1 is completed. The Service-POD (SGW) sends the Update Session to CDL message to the CDL-Endpoint. The session is updated in CDL.

Standards Compliance

This feature complies with the following standards specifications:

- *3GPP TS 29.274 "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3"*

Support for Backoff Timer, Origination TimeStamp, and MaxWait Time

This section describes the support for the Information Elements (IE)—Backoff Timer, Origination Time Stamp, and MaxWait Time.

For more information on technical specifications for the IEs, see *3GPP TS 29.274*.

Backoff Timer

Backoff time indicates the time during which the MME or S4-SGSN refrains from sending the subsequent PDN connection establishment requests to the PGW for the congested APN, for services other than service users or emergency services.

The backoff timer feature enables rejecting new attaches for the failure condition when IP addresses are exhausted.

When IP addresses are exhausted, the PGW-C/SMF detects the status as failure and adds backoff timer in Create Session Response. SGW forwards the backoff timer value to the MME in the Create Session Response.

Origination Time Stamp

Origination Time Stamp is the time at which the originating entity initiated the request. The time stamp is in UTC format.

MME/SGSN and TWAN/ePDG contain the Origination Time Stamp IE on S11/S4 and S2a/S2b interfaces, respectively.

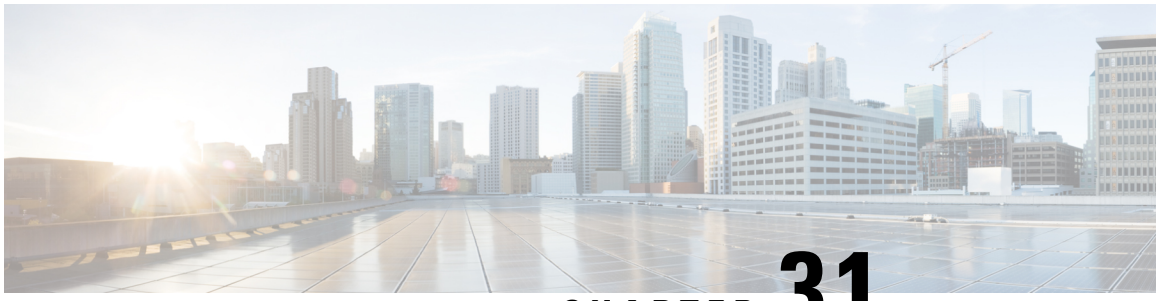
SGW receives the Origination Time Stamp IE from MME/SGSN and includes the IE on the S5/S8 interface.

MaxWaitTime

MaxWaitTime indicates the duration (number of milliseconds since the Origination Time Stamp has lapsed) during which the originator of the request waits for the response.

MME/SGSN and TWAN/ePDG contain the MaxWaitTime IE on S11/S4 and S2a/S2b interfaces, respectively.

SGW contains the MaxWaitTime IE on the S5/S8 interface.



CHAPTER 31

Inter System RAT Handover

- Feature Summary and Revision History, on page 327
- Feature Description, on page 327
- How it Works, on page 328

Feature Summary and Revision History

Summary Data

Table 137: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 138: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

cnSGW-C is the Control Plane Network Functions (NF) of the Converged Core Network (4G-5GC).

cnSGW-C NF is built on top of SMI architecture. cnSGW-C acts as the UE anchor and supports mobility procedures along with session setup and termination procedures as specified in 3GPP TS 23.401, 23.214.

cnSGW-C User Plane (UP) is used to create UP sessions and bearers to carry data traffic.

This feature supports the following procedures in cnSGW-C:

- Wi-Fi to LTE
- GnGp to LTE Hand Over

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows of this feature.

Wi-Fi to LTE Success Call Flow

This section describes the Wi-Fi to LTE success call flow.

Figure 69: Wi-Fi to LTE Success Call Flow

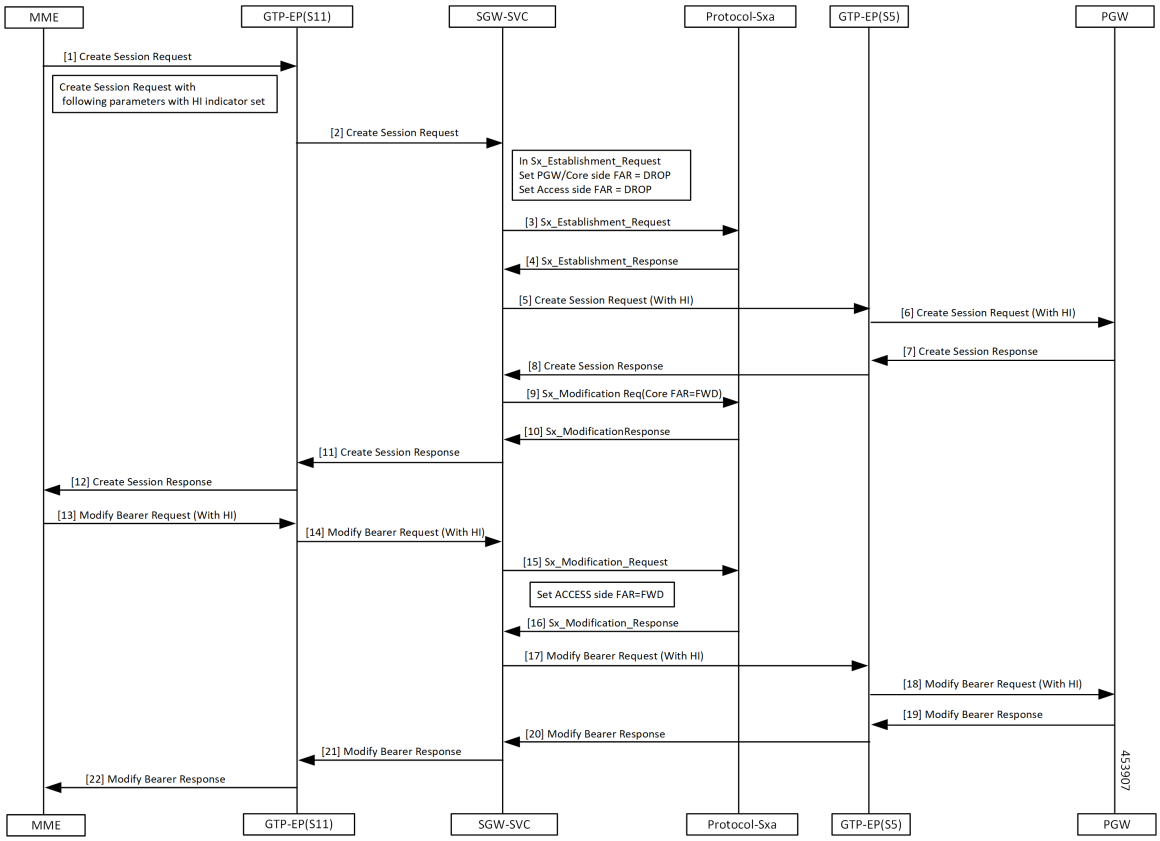


Table 139: Wi-Fi to LTE Success Call Flow Description

Step	Description
1	The MME sends the Create Session Request to the GTP-EP(S11) with: • RAT as EUTRAN • The handoff indicator set to TRUE.
2	The GTP-EP(S11) forwards the Create Session Request to the SGW-SVC.
3	The SGW-SVC sends the Sx Establishment Request to the Protocol-Sxa.
4	The Protocol-Sxa sends the Sx Establishment Response to the SGW-SVC.
5	The SGW-SVC sends the Create Session Request (with HI) to the GTP-EP(S5).
6	The GTP-EP(S5) forwards the Create Session Request (with HI) to the PGW.
7	The PGW sends the Create Session Response to the GTP-EP(S5). The PGW provides IPv6 Prefix.
8	The GTP-EP(S5) forwards the Create Session Response to the SGW-SVC.
9	The SGW-SVC sends the Sx Modification Request to the Protocol-Sxa.
10	The Protocol-Sxa sends the Sx Modification Response to the SGW-SVC.
11	The SGW-SVC sends the Create Session Response to the GTP-EP(S11).
12	The GTP-EP(S11) sends the Create Session Response to the MME.
13	The MME sends the Modify Bearer Request (with HI) to the GTP-EP(S11).
14	The GTP-EP forwards the Modify Bearer Request (with HI) to the SGW-SVC.
15	The SGW-SVC sends the Sx Modification Request to the Protocol-Sxa.
16	The Protocol-Sxa sends the Sx Modification Response to the SGW-SVC.
17	The SGW-SVC forwards the Modify Bearer Request (with HI) to the GTP-EP(S5).
18	The GTP-EP(S5) forwards the Modify Bearer Request (with HI) to the PGW.
19	The PGW sends the Modify Bearer Response to the GTP-EP(S5).
20	The GTP-EP(S5) forwards the Modify Bearer Response to the SGW-SVC.
21	The SGW-SVC forwards the Modify Bearer Response to the GTP-EP(S11).

Step	Description
22	The GTP-EP(S11) forwards the Modify Bearer Response to the MME. The S1 SGW FTEID is the same as the S1-U SGW FTEID sent in Create Session Response from the SGW-SVC to the MME. The SGW-SVC can now send the downlink packets to the eNodeB, and the switching of the data path from Wi-Fi to LTE occurs after the Modify Bearer Response.

GnGp to LTE Handover with OI Indicator Set Call Flow

This section describes the GnGp to LTE Handover with Operation Indication (OI) Indicator Set call flow.

Figure 70: GnGp to LTE Handover with OI Indicator Set Call Flow

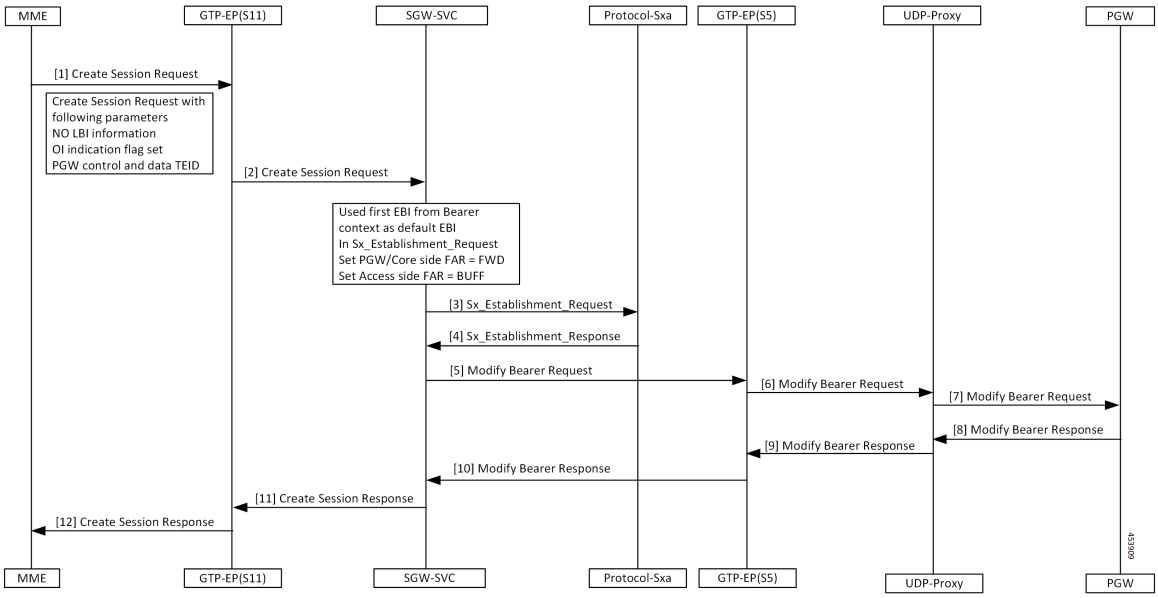


Table 140: GnGp to LTE Handover with OI Indicator Set Call Flow Description

Step	Description
1	The MME sends the Create Session Request to the GTP-EP(S11) with the following information: • EBI List (No LBI Information) • PGW control and data TEID • OI Indicator flag set
2	The GTP-EP(S11) forwards the Create Session Request to the SGW-SVC.
3	The SGW-SVC sends the Sx Session Establishment Request to the Protocol-Sxa.
4	The Protocol-Sxa sends the Sx Establishment Response to the SGW-SVC.
5	The SGW-SVC sends the Modify Bearer Request to GTP-EP(S5).

Step	Description
6	The GTP-EP(S5) forwards the Modify Bearer Request to the UDP-proxy.
7	The UDP-proxy forwards the Modify Bearer Request to the PGW.
8	The PGW sends the Modify Bearer Response with the default EBI information to the UDP-Proxy.
9	The UDP-proxy forwards the Modify Bearer Response to the GTP-EP(S5).
10	The GTP-EP(S5) forwards the Modify Bearer Response to the SGW-SVC.
11	The SGW-SVC sends the Create Session Response with the default EBI information to the GTP-EP(S11).
12	The GTP-EP(S11) forwards the Create Session Response to the MME.

GnGp to LTE Handover with OI Indicator Unset Call Flow

This section describes the GnGp to LTE Handover with Operation Indication (OI) Indicator Unset call flow.

Figure 71: GnGp to LTE Handover with OI Indicator Unset Call Flow

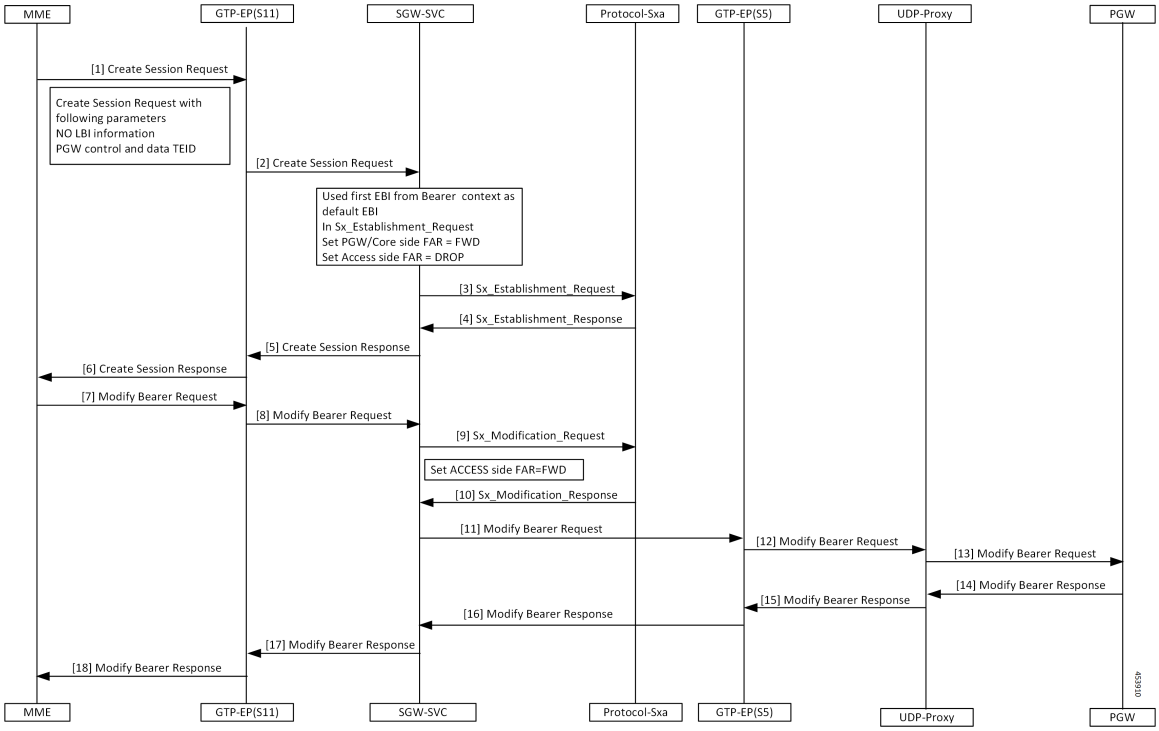


Table 141: GnGp to LTE HO with OI Indicator Unset Call Flow Description

Step	Description
1	The MME sends the Create Session Request to the GTP-EP(S11) with the following information: • EBI List (No LBI Information) • PGW control and data TEID • OI Indicator flag unset
2	The GTP-EP(S11) forwards the Create Session Request to the SGW-SVC.
3	The SGW-SVC sends the Sx Session Establishment Request to the Protocol-Sxa.
4	The Protocol-Sxa sends the Sx Establishment Response to the SGW-SVC.
5	The SGW-SVC sends the Create Session Response to the GTP-EP(S11).
6	The GTP-EP(S11) forwards the Create Session Response to the MME.
7	The MME sends the Modify Bearer Request to the GTP-EP(S11).
8	The GTP-EP(S11) forwards the Modify Bearer Request to the SGW-SVC.
9	The SGW-SVC sends the Sx Modification Request to the Protocol-Sxa.
10	The Protocol-Sxa sends the Sx Modification Response to the SGW-SVC.
11	The SGW-SVC sends the Modify Bearer Request to the GTP-EP(S5).
12	The GTP-EP(S5) forwards the Modify Bearer Request to the UDP-Proxy.
13	The UDP-proxy forwards the Modify Bearer Request to the PGW.
14	The PGW sends the Modify Bearer Response with the default EBI information to the UDP-Proxy.
15	The UDP-Proxy forwards the Modify Bearer Response to the GTP-EP(S5).
16	The GTP-EP(S5) forwards the Modify Bearer Response to the SGW-SVC.
17	The SGW-SVC forwards the Modify Bearer Response to the GTP-EP(S11).
18	The GTP-EP(S11) forwards the Modify Bearer Response to the MME. The S1 SGW FTEID is the same as the S1-U SGW FTEID sent in Create Session Response from the SGW-SVC to the MME. The SGW-SVC can now send the downlink packets to the eNodeB, and the switching of the data path from Wi-Fi to LTE occurs after the Modify Bearer Response.

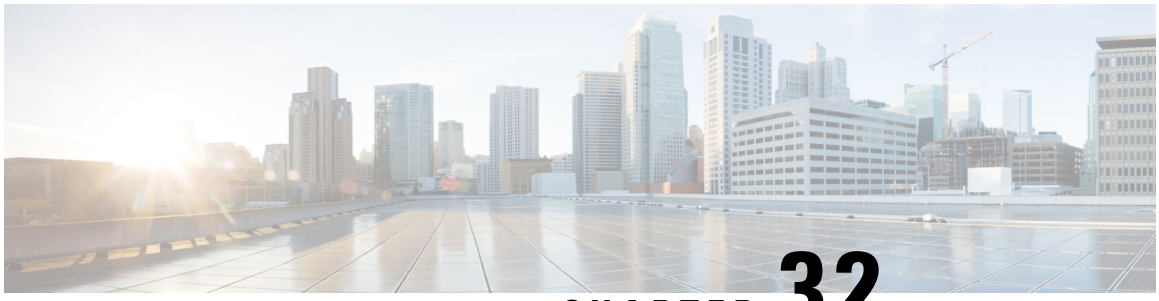
Note cnSGW-C clears the call when the received default EBI in the Modify Bearer Response differs with the first EBI in the following scenarios:

- GnGp to LTE HO with OI Indicator Set
- GnGp to LTE HO with OI Indicator Unset

Standards Compliance

This feature complies with the following standards specifications:

- *3GPP TS 23.401 "General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access"*
- *3GPP TS 23.214 "Architecture enhancements for control and user plane separation of EPC nodes"*



CHAPTER 32

5GS Interworking Suppression Control

- 5GSIWKI suppression control, on page 335
- How 5GSIWKI suppression works, on page 335
- Configure 5GSIWKI suppression, on page 336
- Monitoring Subscriber Information, on page 337

5GSIWKI suppression control

5GSIWKI Suppression Control is a mechanism that allows the cnSGWc to manage the forwarding of the 5GS Interworking Indication (5GSIWKI) flag, which is received from the Mobility Management Entity (MME) during session establishment. This control enables operators to customize how the 5GSIWKI flag is handled and forwarded, enhancing flexibility for interworking scenarios between 4G and 5G networks.

The suppression control feature is a configurable through the command-line interface (CLI). The control mechanism applies logic based on the subscriber's status—whether the subscriber is a home user or a roamer.

When a User Equipment (UE) that supports 5G interworking initiates a session, the MME includes the 5GSIWKI flag in the Create Session Request. The cnSGWc uses the suppression control configuration to decide whether to forward or suppress this flag towards the S5/S8 interface, depending on the subscriber type.

How 5GSIWKI suppression works

Summary

The cnSGWc processes and forwards the 5GSIWKI flag received from the MME based on the configured **ignore-5g-iwk setting** within the subscriber's assigned call-control-profile.

The key components involved in the process are:

- User Equipment (UE): Initiates a session that may support 5G interworking.
- Mobility Management Entity (MME): Includes the 5GS Interworking Indication (5GSIWKI) flag in the Create Session Request.
- Cloud Native Serving Gateway (cnSGW): Receives the Create Session Request and applies suppression logic.

- Session Management Function (SMF)/Packet Gateway (P-GW): Receives the forwarded Create Session Request with the modified 5GSIWKI flag.
- Call-control-profile: A configuration profile that determines the `ignore-5g-iwk` setting for a subscriber.

Workflow

The process involves the following stages:

1. A User Equipment (UE) that supports 5G interworking initiates a session.
2. The MME includes the 5GS Interworking Indication (5GSIWKI) flag as "true" in the Create Session Request and sends it to the cnSGW.
3. The cnSGWc receives the Create Session Request.
4. The cnSGWc identifies the call-control-profile assigned to the subscriber.
5. The cnSGWc checks the ignore-5g-iwk setting within the identified call-control-profile.
   - If `ignore-5g-iwk` is set to `true` (typically for home subscribers), the cnSGWc transmits the 5GSIWKI as `False` towards the S5/S8 interface, regardless of the received value.
   - If `ignore-5g-iwk` is set to `false` (typically for inbound roaming subscribers or default behavior), the cnSGWc forwards the 5GSIWKI as `True` towards the S5/S8 interface if it was received as `True` from the MME.
6. The SMF/PGW receives the Create Session Request with the potentially modified 5GSIWKI flag.

Result

The 5GSIWKI flag is handled according to the operator's policy, allowing for differentiated behavior for home and roaming subscribers. After a system upgrade, the default behavior conforms to the 3GPP specification, where no suppression logic applies, and the 5GSIWKI Indication IE is forwarded as `True` if received as `True`.

Configure 5GSIWKI suppression

To control whether the cnSGWc suppresses the 5GS Interworking Indication (5GSIWKI) flag when forwarding it to the SMF/P-GW.

This configuration allows operators to differentiate 5GSIWKI handling for home and roaming subscribers based on their assigned call-control-profile. The new CLI configuration applies exclusively to new call sessions initiated after the configuration is applied.

Before you begin

Follow these steps to configure 5GSIWKI suppression:

Procedure

Step 1 Create or enter an existing policy call-control-profile in the global configuration mode.

Example:

```
policy call-control-profile profile_name
```

Replace *profile_name* with the desired name for your call control profile (for example, `cc_home`, `cc_roam`).

Step 2 Configure the ignore-5g-iwk setting:

Example:

```
ignore-5g-iwk { false | true }
```

Use **true** to suppress sending the 5GS Interworking Indication towards the S5/S8 interface.

Use **false** to forward the 5GS Interworking Indication as received from the MME. The default value is **false**.

Step 3 Exit the call-control-profile configuration mode and global configuration mode.

Step 4 (Optional): To verify the configuration, use the following show running-config command:

Example:

```
[sgw] smf# show running-config
policy call-control-profile ccp1
  charging-mode           none
  sgw-charging-profile ch1
  ignore-5g-iwk           true
exit
```

The cnSGWc applies the configured 5GSIWKI suppression behavior to new call sessions associated with the specified call-control-profile.

Monitoring Subscriber Information

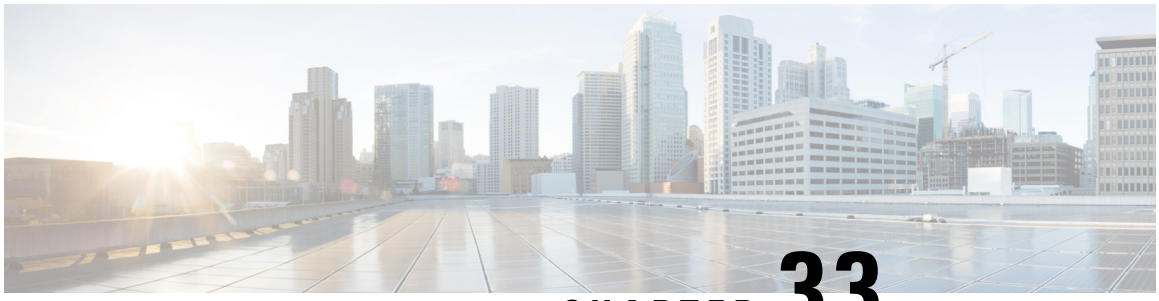
The `show subscriber` command has been enhanced to display the `call-control-profile` used by a subscriber. This helps in verifying which profile is applied to a specific subscriber session.

Sample `show subscriber` Output:

```
[sgw] smf# show subscriber nf-service sgw imsi 123456789012348 gr-instance 1
{
  "subResponses": [
    {
      "status": true,
      "genericInfo": {
        "imsi": "imsi-123456789012348",
        "mei": "imei-123456786666660",
        "msisdn": "msisdn-223310101010101",
        "accessType": "EUTRAN",
        "plmnId": {
          "mcc": "123",
          "mnc": "456"
        },
        "sgwProfileName": "sgw1",
        "unAuthenticatedImsi": "No",
        "subscriberType": "NonVolte"
      },
      "s11cInterfaceInfo": {
        "sgwTeid": "[0x11000a80] 285215360",
        "sgwIPv4Address": "209.165.201.1",
        "mmeTeid": "[0x54e] 1358",
        "mmeIPv4Address": "209.165.201.2"
      },
      "pdnInfoList": {
        "totalPdn": 1,
        "pdnInfo": [
```

```
{
  "pdnId": "PDN-1",
  "apn": "intershat",
  "attachType": "Initial Attach",
  "sgwRelocState": "N/A",
  "operatorPolicyName": "opPol1",
  "dnnProfileName": "intershat",
  "defaultEbi": 5,
  "pdnType": "IPv4",
  "allocatedIPv4": "209.165.201.3",
  "apnSelectionMode": "Subscribed",
  "ambrUplink": "10 Kbps",
  "ambrDownlink": "20 Kbps",
  "s5cInterfaceInfo": {
    "sgwTeid": "[0x51000a80] 1358957184",
    "sgwIPv4Address": "209.165.201.1",
    "pgwTeid": "[0x3ea] 1002",
    "pgwIPv4Address": "209.165.201.4"
  },
  "sxaInterfaceInfo": {
    "selectedUP": "209.165.200.225",
    "upEpKey": "209.165.201.2:209.165.200.233",
    "cpSeid": "[0x11000a8051000a80] 1224990644875823744",
    "upSeid": "[0x3e9] 1001"
  },
  "bearerInfoList": {
    "totalBearer": 1,
    "bearerInfo": [
      {
        "bearerId": "Bearer-1",
        "state": "Connected",
        "ebi": 5,
        "isDefaultBearer": true,
        "qosInfo": {
          "qci": 6,
          "arp": 113
        },
        "s1uInterfaceInfo": {
          "sgwTeid": "[0x550] 1360",
          "sgwIPv4Address": "209.165.200.238",
          "eNodeBTeid": "[0x551] 1361",
          "eNodeBIPv4Address": "209.165.200.239"
        },
        "s5uInterfaceInfo": {
          "sgwTeid": "[0x54f] 1359",
          "sgwIPv4Address": "209.165.201.10",
          "pgwTeid": "[0x3eb] 1003",
          "pgwIPv4Address": "209.165.201.12"
        },
        "chargingId": 303174163
      }
    ]
  },
  "uli": {
    "mcc": "123",
    "mnc": "456",
    "tac": "0x92a",
    "eci": "0x12d687"
  },
  "uetimeZone": {
    "timeZone": "+0:15",
    "dayLightSavingTime": "+1 hour"
  },
  "plmnType": "VISITOR",
```

```
            "collocatedSub": "NonCollocated",
            "dataTunnelType": "IPV4",
            "ipSrcViolation": "N/A",
            "callControlProfileName": "ccp1"
          }
        ]
      }
    }
  ]
}
```



CHAPTER 33

Intra-MME and Inter-MME Handover Procedures

- Feature Summary and Revision History, on page 341
- Feature Description, on page 341
- How it Works, on page 342
- Intra-MME and Inter-MME Handover Procedures OAM Support, on page 349

Feature Summary and Revision History

Summary Data

Table 142: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 143: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

cnSGW-C supports Intra-MME Intra-SGW, and Inter-MME Intra-SGW handover.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Inter-MME Handover Active-Active Transition Call Flow

This section describes the Inter-MME Handover Active-Active Transition call flow.

Figure 72: Inter-MME Handover Active-Active Transition Call Flow

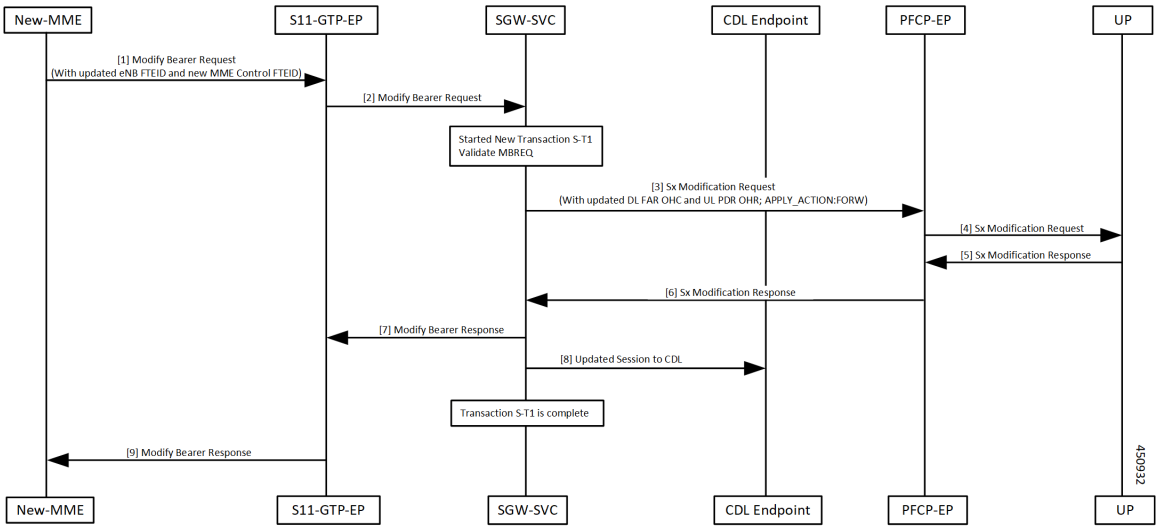


Table 144: Inter-MME Handover Active-Active Transition Call Flow Description

Step	Description
1	New-MME sends the Modify Bearer Request to the S11-GTP-EP pod, with updated eNodeB F-TEID and new MME control F-TEID.
2	S11-GTP-EP pod forwards the Modify Bearer Request to the SGW-SVC. SGW-SVC performs the following: • Creates a new transaction S-T1 • Validates the Modify Bearer Request
3	SGW-SVC sends the the Sx Modification Request with downlink FAR OHC, uplink PDR OHR, and APPLY ACTION as FORWARD, to the PFCP-EP pod.
4	PFCP-EP pod sends the Sx Modification Request to the UP.
5	UP sends the Sx Modification Response to the PFCP-EP pod.

Step	Description
6	PFCP-EP pod sends the Sx Modification Response to the SGW-SVC.
7	SGW-Service pod sends the Modify Bearer response to the S11-GTP-EP pod.
8	SGW-SVC sends the Updated Session to the CDL Endpoint. Transaction S-T1 is complete.
9	S11-GTP-EP pod sends the Modify Bearer Response to the New-MME.

Intra-MME Handover Active-Active Transition Call Flow

This section describes the Inear-MME Handover Active-Active Transition call flow.

Figure 73: Intra-MME Handover Active-Active Transition Call Flow

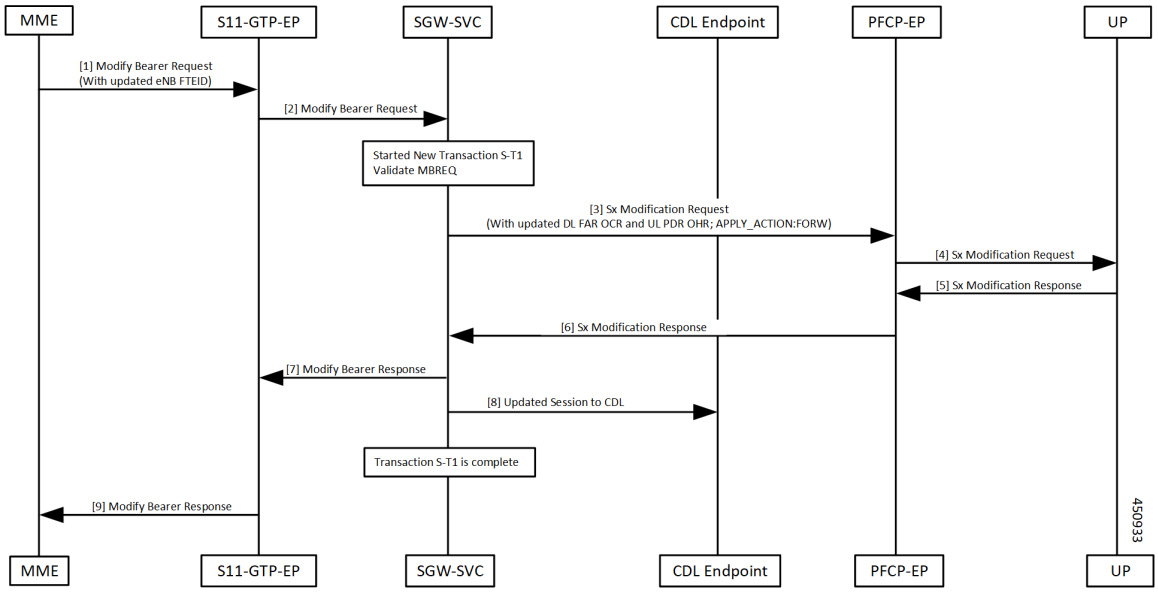


Table 145: Intra-MME Handover Active-Active Transition Call Flow Description

Step	Description
1	MME sends the Modify Bearer Request to the S11-GTP-EP pod, with the updated eNodeB F-TEID.
2	S11-GTP-EP pod sends the Modify Bearer Request to the SGW-SVC. SGW-SVC performs the following: • Creates a new transaction S-T1 • Validates Modify Bearer Request
3	SGW-SVC sends the the Sx Modification Request with downlink FAR OCR, uplink PDR OHR, and APPLY ACTION as FORWARD, to the PFCP-EP pod.

Step	Description
4	PFCP-EP pod forwards the Sx Modification Request to the UP.
5	UP sends the Sx Modification Response to the PFCP-EP pod.
6	PFCP-EP pod sends the Sx Modification Response to the SGW-SVC.
7	SGW-SVC sends the Modify Bearer Response to the S11-GTP-EP pod.
8	SGW-SVC sends the Updated Session to the CDL Endpoint. Transaction S-T1 is complete.
9	S11-GTP-EP pod sends the Modify Bearer Response to the MME.

Inter/Intra-MME Handover Idle-Idle Transition Call Flow

This section describes the Inter/Intra-MME Handover Idle-Idle Transition call flow.

Figure 74: Inter/Intra-MME Handover Idle-Idle Transition Call Flow

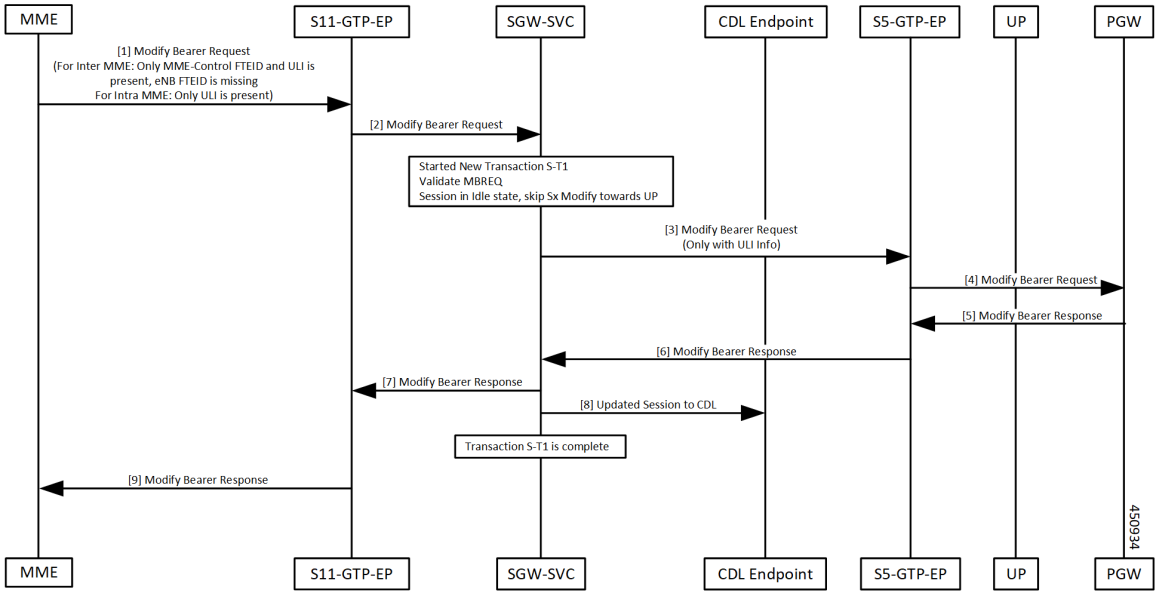


Table 146: Inter/Intra-MME Handover Idle-Idle Transition Call Flow Description

Step	Description
1, 2	For inter-MME, received Modify Bearer request at SGW-Service POD via S11-GTPC-EP POD with MME control F-TEID and ULI. There is no eNodeB F-TEID. For intra-MME, received Modify Bearer request at SGW-Service POD with only ULI present: • Create a new transaction S-T1. • Validate Modify Bearer request. • Skip Sx modification as session is in idle state.

Step	Description
3, 4	Received Modify Bearer request at S5 GTPC-EP POD from SGW-Service POD with ULI. Modify Bearer request is forwarded to PGW.
5, 6	Received Modify Bearer response at S5 GTPC-EP POD from PGW. Modify Bearer response received at SGW-Service POD.
7	SGW-Service POD forwards Modify Bearer response to S11 GTPC-EP POD ingress.
8, 9	Session updated at CDL. Transaction S-T1 is complete. S11 GTPC-EP POD forwards Modify Bearer response to MME.

Inter/Intra-MME Handover Active-Idle Transition Call Flow

This section describes the Inter/Intra-MME Handover Active-Idle transition call flow.

Figure 75: Inter/Intra-MME Handover Active-Idle Transition Call Flow

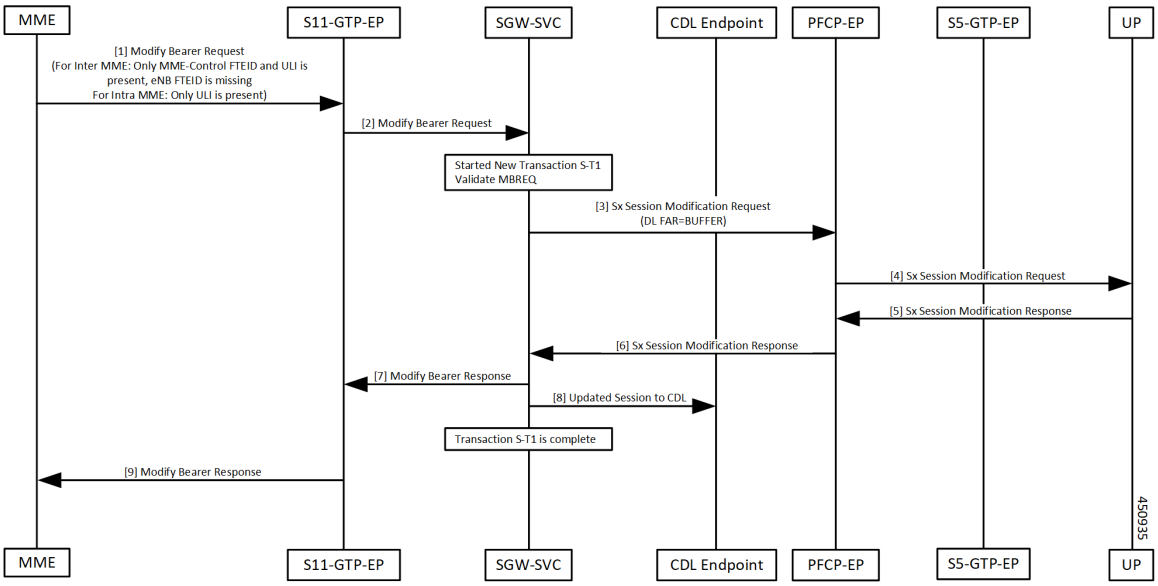


Table 147: Inter/Intra-MME Handover Active-Idle Transition Call Flow Description

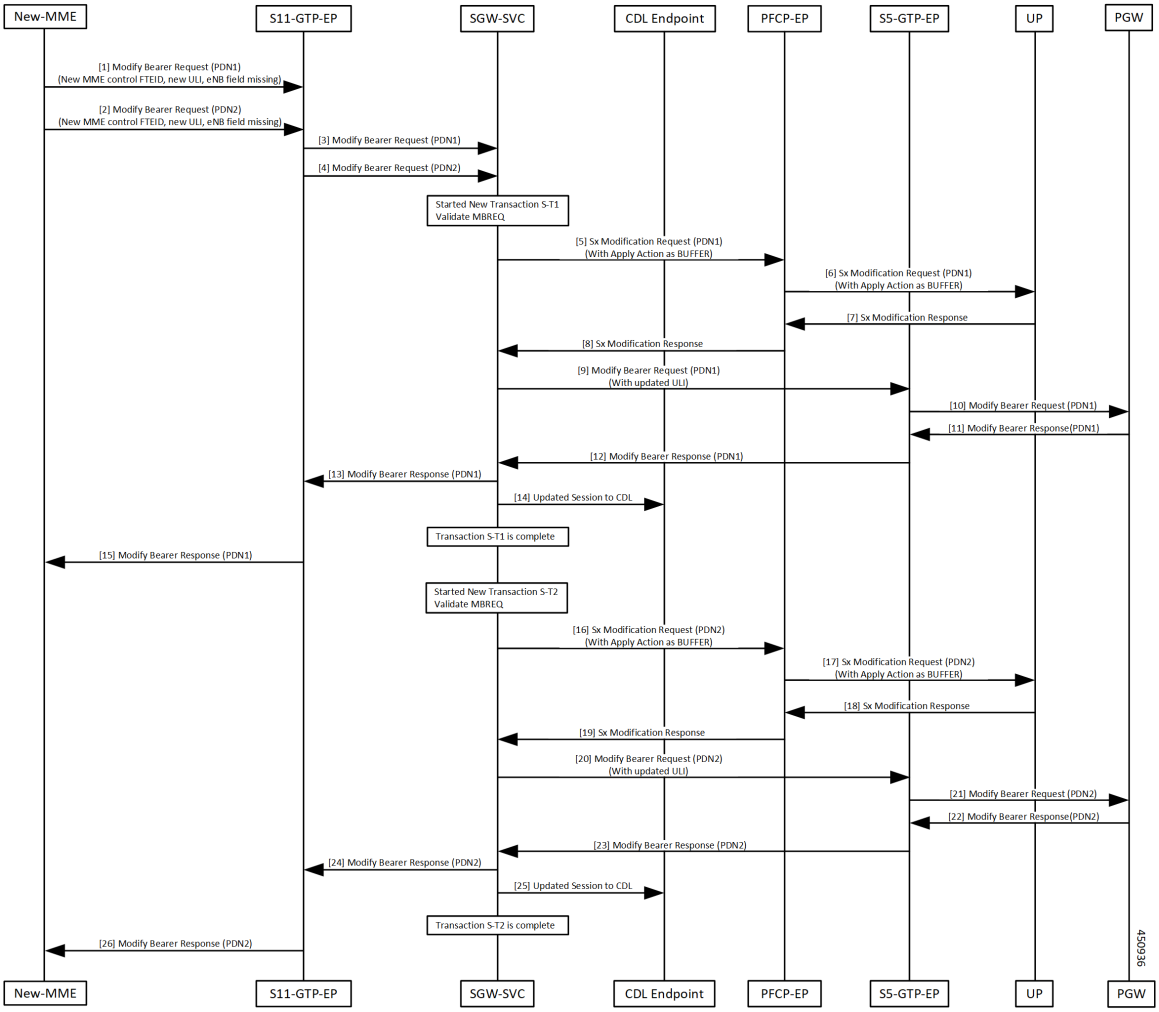
Step	Description
1, 2	For inter-MME, received Modify Bearer request at SGW-Service POD via S11-GTPC-EP POD with MME control F-TEID and ULI. There is no eNodeB F-TEID. For intra-MME, received Modify Bearer request at SGW-Service POD with only ULI present: • Create a new transaction S-T1. • Validate Modify Bearer request.
3, 4	Received Sx Modification request at PFCP-EP POD from SGW-Service POD with downlink FAR as BUFFER. Sx Modification request is forwarded to UP.

Step	Description
5, 6	Received Sx Modification response at at PFCP-EP POD from UP. Sx Modification response received at SGW-Service POD.
7	SGW-Service POD forwards Modify Bearer response to GTPC-EP POD ingress.
8, 9	Session updated at CDL. Transaction S-T1 is complete. GTPC-EP POD forwards Modify Bearer response to MME.

Inter-MME Handover and Multi-PDN Handling Active-Idle Transition with ULI Change Call Flow

This section describes the Inter-MME Handover and Multi-PDN Handling Active-Idle transition with ULI change call flow.

Figure 76: Inter-MME Handover and Multi-PDN Handling Active-Idle Transition with ULI Change Call Flow



Repeat the steps provided in the Table 148: Inter-MME Handover and Multi-PDN Handling Active-Idle Transition with ULI Change Call Flow Description, on page 347 for PDN1 and PDN2 with respective transaction S-T1 and S-T2.

Table 148: Inter-MME Handover and Multi-PDN Handling Active-Idle Transition with ULI Change Call Flow Description

Step	Description
1, 2, 3, 4	Received Modify Bearer request for both the PDNs (PDN1 and PDN2) at SGW-Service POD with new MME control F-TEID and new ULI. There is no eNodeB F-TEID present in Modify Bearer request. • Create a new transaction S-T1. • Validate Modify Bearer request.
5, 6	Received Sx Modification request from SGW-Service POD > PFCP-EP POD with APPLY ACTION as BUFFER. Sx Modification request is forwarded to UP.
7, 8	Received Sx Modification response from UP > PFCP-EP POD. Sx Modification response received at SGW-Service POD.
9, 10	Received Modify Bearer request from SGW-Service POD > S5 GTPC-EP POD with updated ULI. Modify Bearer request is received at PGW.
11, 12	Received Modify Bearer response from PGW > S5 GTPC-EP POD. Modify Bearer response received from S5 GTPC-EP POD > SGW-Service POD.
13	SGW-Service POD forwards Modify Bearer response to GTPC-EP POD ingress.
14, 15	Session updated at CDL. Transaction S-T1 is complete. GTPC-EP POD forwards Modify Bearer response to MME.

Inter-MME Handover with Bearer Context Marked for Removal Call Flow

This section describes the Inter-MME Handover with Bearer Context Marked for Removal call flow.

Figure 77: Inter-MME Handover with Bearer Context Marked for Removal Call Flow

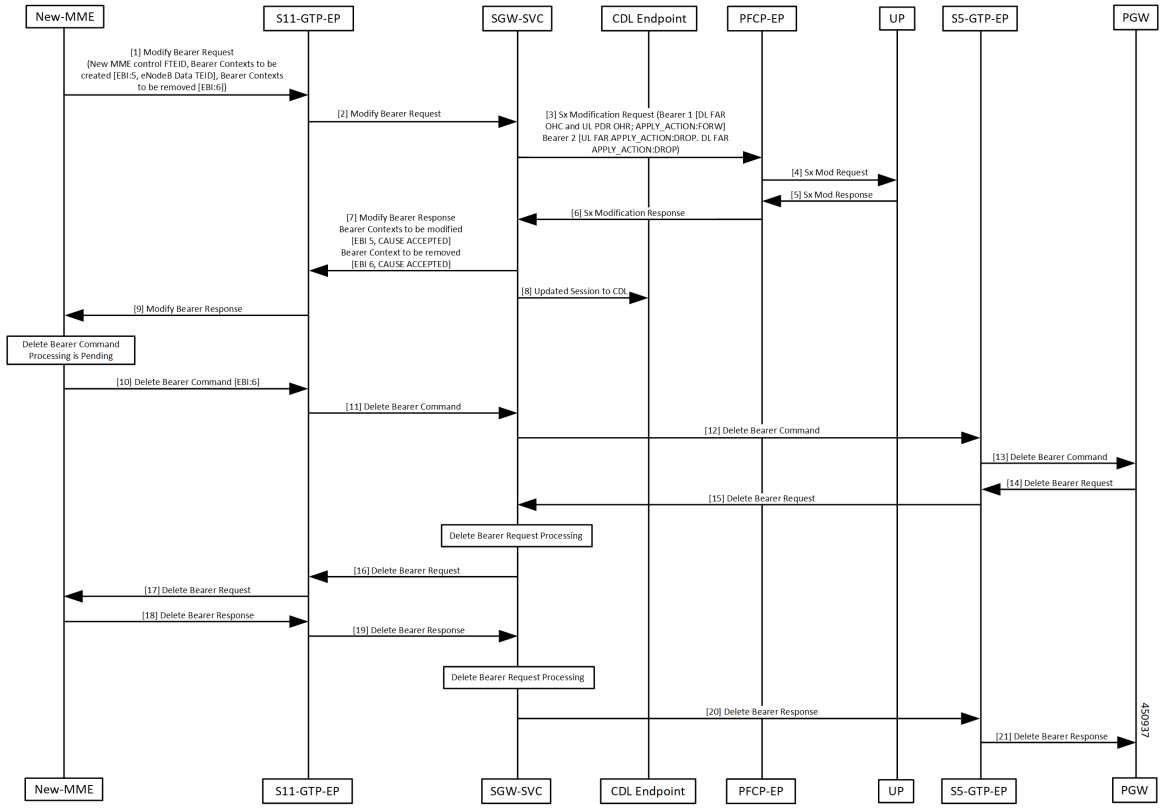


Table 149: Inter-MME Handover with Bearer Context Marked for Removal Call Flow Description

Step	Description
1, 2	Received Modify Bearer request at SGW-Service POD with new MME control F-TEID, bearer-context-1 to be created with EBI:5, eNodeB data TEID, bearer-context-2 to be removed with EBI:6. • Create a new transaction S-T1. • Validate Modify Bearer request.
3, 4	Received Sx Modification request from SGW-Service POD to PFCP-EP POD for: • bearer-context-1: downlink FAR OHC, uplink PDR OHR and APPLY ACTION as FORWARD • bearer-context-2: uplink FAR APPLY ACTION as DROP and downlink FAR APPLY ACTION as DROP Sx Modification request is forwarded to UP.
5, 6	Received Sx Modification response from UP > PFCP-EP POD. Sx Modification response is received at SGW-Service POD.

Step	Description
7	SGW-Service POD forwards Modify Bearer response to GTPC-EP POD ingress with bearer context to be modified (EBI:5, cause as ACCEPTED) and bearer context to be removed (EBI:6, cause as ACCEPTED).
8, 9	Session updated at CDL. GTPC-EP POD forwards Modify Bearer response to MME.
10, 11	Received delete bearer command at SGW-Service POD with EBI:6.
12, 13	SGW-Service POD forwards delete bearer command to S5 GTPC-EP POD. Delete bearer command received at PGW.
14, 15	Received Delete Bearer request from PGW > S5 GTPC-EP POD. Delete Bearer request received at SGW-Service POD.
16, 17	SGW-Service POD processes Delete Bearer request and forwards it to GTPC-EP POD ingress.
18, 19	Delete Bearer response received at SGW-Service POD and processed.
20, 21	Delete Bearer response received at S5 GTPC-EP POD. Delete Bearer response is received at PGW.

Intra-MME and Inter-MME Handover Procedures OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

The following statistics are supported for the Intra-MME and Inter-MME Handover Procedures feature.

Intra-MME Handover

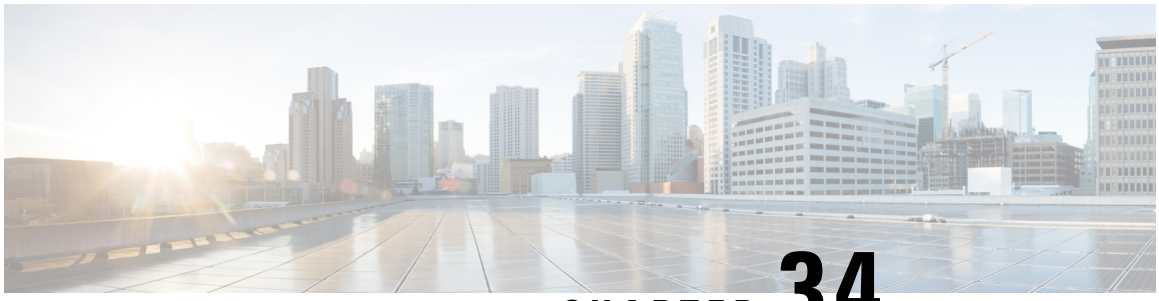
```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="intra_mme_handover",status="attempted",sub_fail_reason=""} 2
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="intra_mme_handover",status="success",sub_fail_reason=""} 2
```

Inter-MME Handover

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="inter_mme_handover",status="attempted",sub_fail_reason=""} 2
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",
instance_id="0",interface="interface_sgw_ingress",reject_cause="",service_name="sgw-service",
sgw_procedure_type="inter_mme_handover",status="success",sub_fail_reason=""} 2
Perform S1 Based SGW handover (with OI=0)
```



CHAPTER 34

MCC/MNC Configuration in the SGW Service

- Feature Summary and Revision History, on page 351
- Feature Description, on page 351
- How it Works, on page 352
- Configuring the MCC or the MNC in the SGW Service , on page 353
- OAM Support, on page 354

Feature Summary and Revision History

Summary Data

Table 150: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 151: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

This feature supports the list of MCC and MNC configuration as a PLMN-list in the SGW profile.

As per the PLMN-list configuration in the SGW profile, the PLMN-type is identified as one of the following:

- Homer
- Roamer
- Visitor

Note If this feature is not enabled, the PLMN-type subscriber is marked as a Visitor, by default.

How it Works

This section describes how the feature works.

Call Flows

This section describes the key call flows for this feature.

PLMN-type Detection Call Flow

This section describes the PLMN-type Detection call flow.

Figure 78: PLMN-type Detection Call Flow

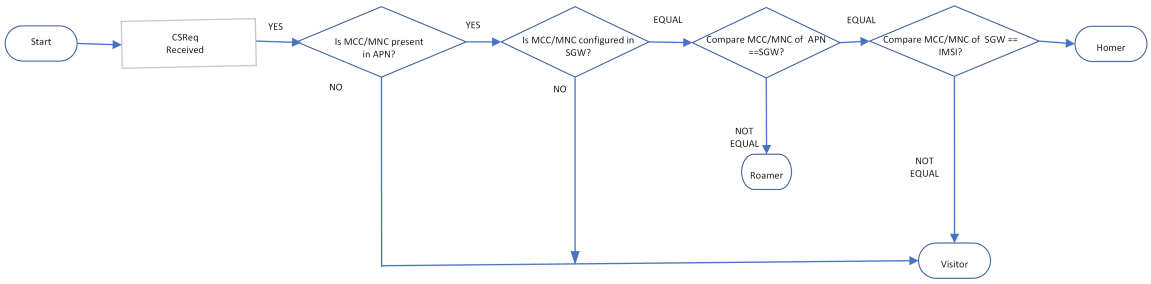


Table 152: PLMN-type Detection Call Flow Description

Step	Description
1	The PLMN-type detection process begins.
2	The Create Session Request is received.
3	The MCC or the MNC presence in the APN is verified: • If YES, the next step is executed. • If NO, the PLMN-type subscriber is concluded as Visitor.

Step	Description
4	The MCC or the MNC presence in the SGW is verified: • If YES, the next step is executed. • If NO, the PLMN-type subscriber is concluded as Visitor.
5	The values of the MCC or the MNC in the APN and the SGW are compared: • If EQUAL, it proceeds to the next step. • If NOT EQUAL, the PLMN-type subscriber is concluded as Roamer.
6	The values of the MCC or the MNC in the SGW and the IMSI are compared: • If EQUAL, the PLMN-type subscriber is concluded as Homer. • If NOT EQUAL, the PLMN-type subscriber is concluded as Visitor.

Configuring the MCC or the MNC in the SGW Service

This section describes how to configure the MCC or the MNC in the SGW service.

Use the following commands to configure the MCC or the MNC in the SGW service.

```
config
   profile sgw sgw_name
      plmn-list
         mcc mcc_value
         mnc mnc_value
         end
```

NOTES:

- **plmn-list**—List of MCC and MNC values.
- **mcc** *mcc_value*—Specify the MCC value. Must be a three-digit number. Example: 123
- **mnc** *mnc_value*—Specify the MNC value. Must be a two or three-digit number. Example: 23 or 456

Configuration Example

The following is an example configuration.

```
config
   profile sgw sgw_1
      plmn-list
         mcc 123
         mnc 456
         end
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the MCC and MNC Configuration in the SGW Service feature.

Active PDN Counters

```
sgw_pdn_counters{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="homer",pdn_type="ipv6",rat_type="EUTRAN",
service_name="sgw-service"} 12
```

```
sgw_pdn_counters{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="roamer",pdn_type="ipv4v6",rat_type="EUTRAN",
service_name="sgw-service"} 3
```

```
sgw_pdn_counters{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="visitor",pdn_type="ipv4",rat_type="EUTRAN",
service_name="sgw-service"} 2
```

Setup or Released PDN Statistics

```
sgw_pdn_stats{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="homer",pdn_type="ipv6",rat_type="EUTRAN",
service_name="sgw-service",status="release"} 1
```

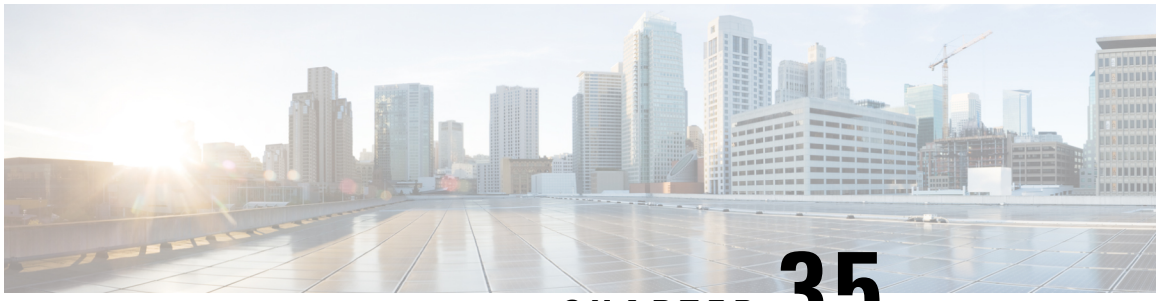
```
sgw_pdn_stats{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="homer",pdn_type="ipv6",rat_type="EUTRAN",
service_name="sgw-service",status="setup"} 13
```

```
sgw_pdn_stats{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="roamer",pdn_type="ipv4v6"rat_type="EUTRAN",
service_name="sgw-service",status="release"} 1
```

```
sgw_pdn_stats{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="roamer",pdn_type="ipv4v6",rat_type="EUTRAN",
service_name="sgw-service",status="setup"} 4
```

```
sgw_pdn_stats{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="visitor",pdn_type="ipv4",rat_type="EUTRAN",
service_name="sgw-service",status="release"} 1
```

```
sgw_pdn_stats{app_name="smf",cluster="cn",data_center="cn",
instance_id="0",pdn_plmn_type="visitor"pdn_type="ipv4",rat_type="EUTRAN",
service_name="sgw-service",status="setup"} 3
```



CHAPTER 35

Message Interactions Support

- Feature Summary and Revision History, on page 357
- Feature Description, on page 358
- How it Works, on page 359

Feature Summary and Revision History

Summary Data

Table 153: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled – Always-on
Related Documentation	Not Applicable

Revision History

Table 154: Revision History

Revision Details	Release
Procedures added for: • Collision Resolver • Multiple CBR • Double Delete Optimized • Abort Handling of Low priority and Handling Suspension	2021.02.0

Revision Details	Release
First introduced.	2021.01.0

Feature Description

The Message Interactions feature provides the capability to receive and process the messages from different peers (UPF, MME, and PGW), and performs the priority resolution.

The following are the examples of message interaction scenarios and priorities:

- The Modify Bearer Request (MBR) and Update Bearer Request (UBR) received for the same PDN1 waits for the Sx Modify Response from UPF. The UBR is processed after the MBR is completed. In this scenario, the UBR process is suspended until the MBR is processed.
- The UBR1 received for the PDN1 while processing a Release Access Bearer (RAB) for the same PDN1. The UBR is processed after the RAB is completed. In this scenario, the UBR1 process is suspended when the RAB procedure is in progress.
- The existing PDN procedure is stopped when the disconnect procedure (Delete Session Request (DSR), Delete Bearer Request (DBR), or Clear Sub) is sent for the same PDN. For example, cnSGW-C receives the DSR for the PDN when the CBR1 and UBR1 procedures are in progress for the same PDN1. The DSR processing is started, and CBR1 and UBR1 processing is stalled. For more information, see the Graceful Stop the Existing PDN Procedure Call Flow, on page 362 call flow.
- The existing UE procedure (RAB or DDN) is stopped when the disconnect procedure (DSR, DBR, or Clear Sub) for the PDN is received. For example, cnSGW-C receives the DBR for the PDN while processing the RAB or DDN. The DBR procedure is started, and the RAB or DDN procedure is stopped.
- The incoming procedure for the PDN is stopped when the disconnect procedure (DSR, DBR, or Clear Sub) for the same PDN is in progress. For example, the UBR receives the PDN when sending the DSR for the same PDN. The UBR procedure is stopped, and the DSR procedure continues.
- The new incoming UE procedure is stopped when processing the disconnect procedure (DSR, DBR, or Clear Sub) for the same PDN. For example, the RAB received for the PDN1 when processing the multi-PDN call DSR for the same PDN1. The RAB procedure is stopped and rescheduled after the DSR for PDN1 gets completed.
- The CBR and the UBR message handling are stopped when the initial attach procedure is in progress.
- Optimization of the double delete handling. For example, the cnSGW-C receives the DSR from MME and DBR in the PGW, for which the DBR Sx modify step is pending toward the UPF. The DBR signaling is not initiated toward the S11 interface.
- The processing of the low priority procedures is stopped when the high priority procedure is received on the same bearer. For example, cnSGW-C receives the DBR for the PDN on a dedicated bearer while processing UBR on the same bearer. The DBR handling procedure is started, and the processing of the existing UBR procedure on the same bearer is stopped.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

CBR Multi-PDN Call Flow

This section describes the CBR Multi-PDN call flow.

Figure 79: CBR Multi-PDN Call Flow

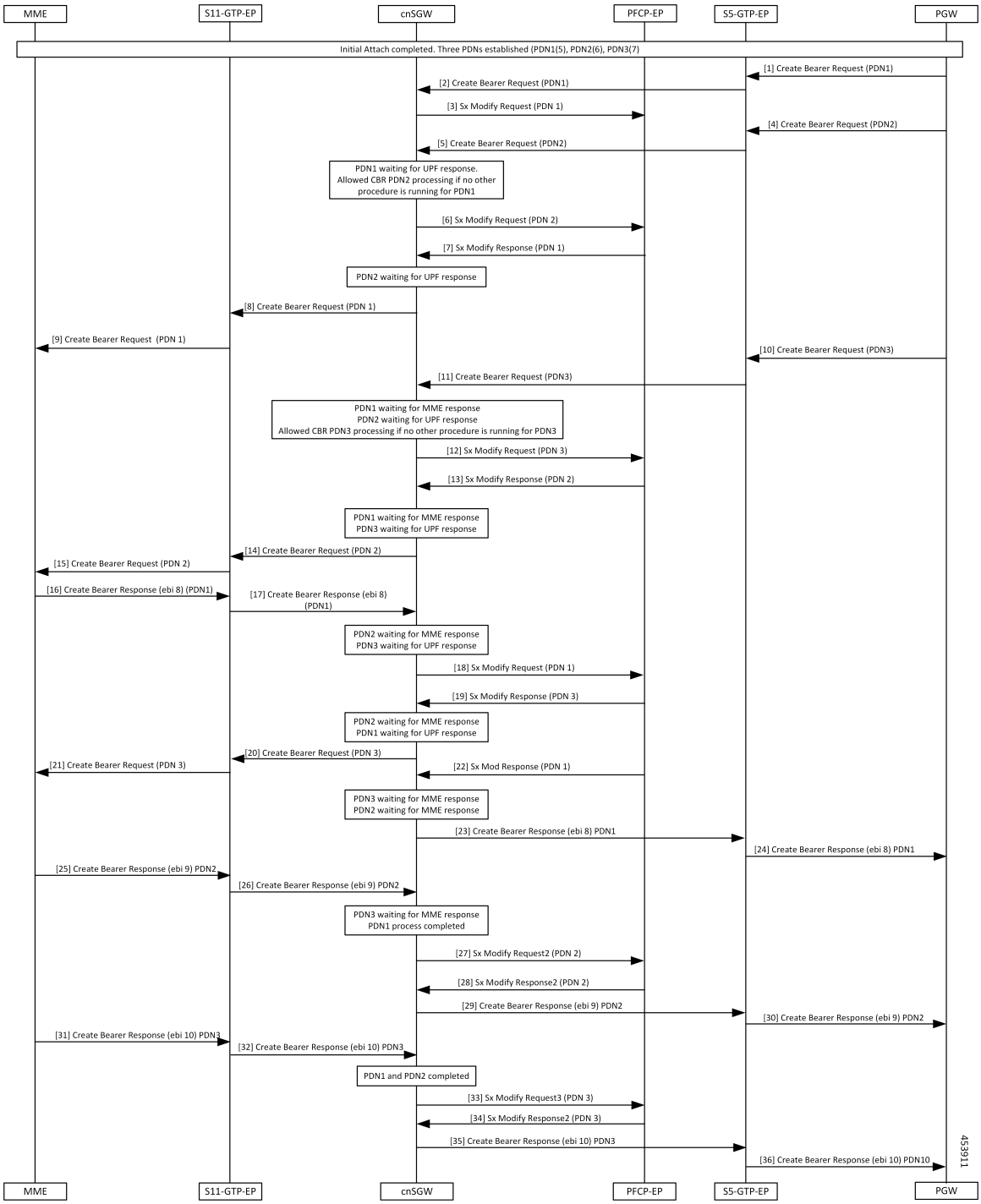


Table 155: CBR Multi-PDN Call Flow Description

Step	Description
1	The Initial Attach process is completed with the three established PDNs. The PGW sends a Create Bearer Request for PDN1 to the S5-GTP-EP.
2	The S5-GTP-EP sends the Create Bearer Request for PDN1 to the cnSGW.
3	The cnSGW sends a Sx Modify Request for PDN1 to the PFCP-EP.
4	The PGW sends a Create Bearer Request for PDN2 to the S5-GTP-EP.
5	The S5-GTP-EP sends the Create Bearer Request PDN2 to the cnSGW.
6	The cnSGW sends a Sx Modify Request to PFCP-EP for PDN2.
7	The PFCP-EP sends the Sx Modify Response for PDN1 to the cnSGW.
8	The cnSGW sends the Create Bearer Request for PDN1 to the S11-GTP-EP.
9	The S11-GTP-EP sends the Create Bearer Request for PDN1 to the MME.
10	The PGW sends the Create Bearer Request (PDN3) to the S5-GTP-EP.
11	The S5-GTP-EP sends the Create Bearer Request (PDN3) to the cnSGW.
12	The cnSGW sends the Sx Modify Request (PDN3) to the PFCP-EP.
13	The PFCP-EP sends the Sx Modify Response for PDN2 to the cnSGW.
14	The cnSGW sends the Create Bearer Request for PDN2 to the S11-GTP-EP.
15	The S11-GTP-EP sends the Create Bearer Request for PDN1 to the MME.
16	The MME sends the Create Bearer Response for PDN1 to the S11-GTP-EP.
17	The S11-GTP-EP sends the Create Bearer Response for PDN1 to the cnSGW.
18	The cnSGW sends the Sx Modify Request for PDN1 to the PFCP-EP.
19	The PFCP-EP sends the Sx Modify Response for PDN3 to the cnSGW.
20	The cnSGW sends the Create Bearer Request for PDN3 to the S11-GTPC-EP.
21	The S11-GTPC-EP sends the Create Bearer Request for PDN3 to the MME.
22	The PFCP-EP sends the Sx Modify Response2 for PDN1 to the cnSGW.
23	The cnSGW sends the Create Bearer Response for PDN1 to the S5-GTP-EP.
24	The S5-GTP-EP sends the Create Bearer Response for PDN1 to the PGW.
25	The MME sends the Create Bearer Response for PDN2 to the S11-GTPC-EP.
26	The S11-GTPC-EP sends the Create Bearer Response for PDN1 to the cnSGW.

Step	Description
27	The cnSGW sends the Sx Modify Request to the PFCP-EP for PDN2.
28	The PFCP-EP sends the Sx Modify Response for PDN2 to the cnSGW.
29	The cnSGW sends the Create Bearer Response for PDN2 to the S5-GTP-EP.
30	The S5-GTP-EP sends the Create Bearer Response for PDN2 to the PGW.
31	The MME sends the Create Bearer Response for PDN3 to the S11-GTPC-EP.
32	The S11-GTPC-EP sends the Create Bearer Response for PDN3 to the cnSGW.
33	The cnSGW sends the Sx Modify Request for PDN2 to the PFCP-EP.
34	The PFCP-EP sends the Sx Modify Response for PDN3 to the cnSGW.
35	The cnSGW sends the Create Bearer Response for PDN3 to the S5-GTP-EP.
36	The S5-GTP-EP sends the Create Bearer Response for PDN3 to the PGW.

Graceful Stop the Existing PDN Procedure Call Flow

This section describes the Graceful Stop the Existing PDN Procedure call flow.

Figure 80: Graceful Stop the Existing PDN Procedure Call Flow

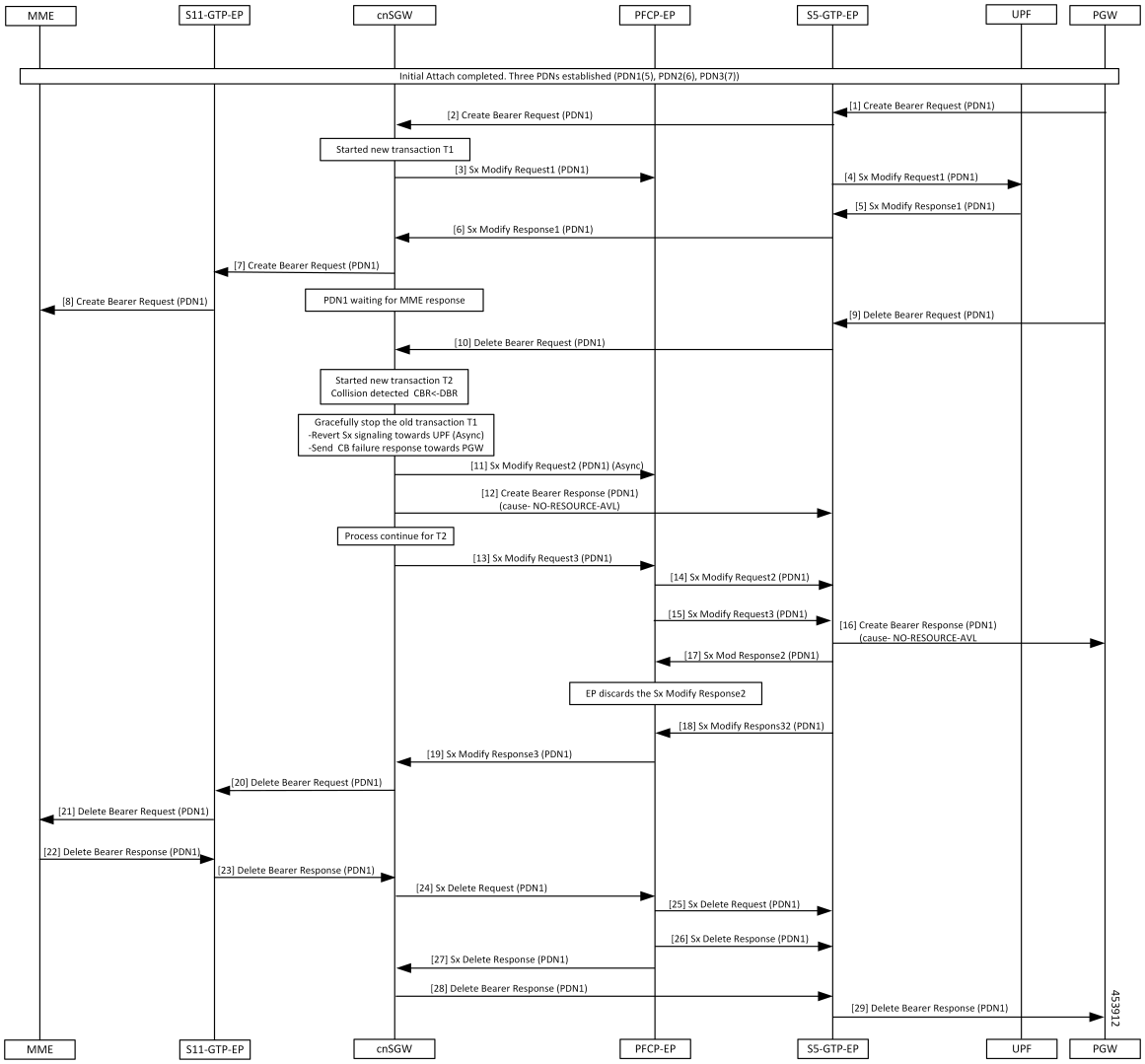


Table 156: Graceful Stop the Existing PDN Procedure Call Flow Description

Step	Description
1	The Initial Attach process is completed with the already established three PDNs. The PGW sends the Create Bearer Request to the S5-GTPC-EP.
2	The S5-GTCP-EP forwards the Create Bearer Request for PDN1 to the cnSGW.
3	The cnSGW forwards the Sx Modify Request for PDN1 to the PFCP-EP. The cnSGW waits for the Sx Modify Response for PDN1.
4	The PFCP-EP sends the Sx Modify Request for PDN1 to the UPF.
5	The UPF sends the Sx Modify Response for PDN1 to the PFCP-EP.

Step	Description
6	The PFCP-EP sends the Sx Modify Response for PDN1 to the cnSGW.
7	The cnSGW sends the Create Bearer Request for PDN1 to the S11-GTPC-EP.
8	The S11-GTPC-EP forwards the Create Bearer Request for PDN1 to the MME.
9	The PGW sends the Delete Bearer Request for PDN1 to the S5-GTPC-EP.
10	The S5-GTPC-EP forwards the Delete Bearer Request for PDN1 to the cnSGW. The cnSGW waits for the MME response and the cnSGW receives the Delete Bearer Request for PDN1. When collision is detected for PDN1, stop the old transaction T1 for PDN1.
11	The cnSGW sends the Sx Modify Request (async) to the PFCP-EP.
12	The cnSGWcnSGW sends the Create Bearer Response with cause No-Resource-Available for PDN1 to the S5-GTPC-EP.
13	The cnSGW sends the Sx Modify Request for DBR to the PFCP-EP.
14	The PFCP-EP forwards the Sx Modify Request for CBR to the UPF.
15	The PFCP-EP forwards the Sx Modify Request for DBR to the UPF.
16	The S5-GTPC-EP sends the Create Bearer Response with cause No.
17	The UPF sends the Sx Modify Response for CBR to the PFCP-EP. The PFCP-EP discards this response.
18	The UPF sends the Sx Modify Response for DBR to the PFCP-EP.
19	The PFCP-EP forwards the Sx Modify Response for DBR to the cnSGW.
20	The cnSGW sends the Delete Bearer Request for PDN1 to the S11-GTPC-EP.
21	The S11-GTPc-EP forwards the Delete Bearer Request for PDN1 to the MME.
22	The MME sends the Delete Bearer Response for PDN1 to the S11-GTPC-EP.
23	The S11-GTCP-EP forwards the Delete Bearer Response for PDN1 to the cnSGW.
24	The cnSGW sends the Sx Delete Request for PDN1 to the PFCP-EP.
25	The PFCP-EP forwards the Sx Delete Request for PDN1 to the UPF.
26	The UPF sends the Sx Delete Response for PDN1 to the PFCP-EP.
27	The PFCP-EP forwards the Sx Delete Response for PDN1 to the cnSGW.
28	The cnSGW sends the Delete Bearer Response for PDN1 to the S5-GTPC-EP.
29	The S5-GTPC-EP forwards the Delete Bearer Response for PDN1 to the PGW.

Inter MME Handover with Multi-PDN Handling (With PGW Interaction) Call Flow

This section describes the Inter MME Handover with Multi-PDN Handling (With PGW Interaction) call flow.

Figure 81: Inter MME Handover with Multi-PDN Handling (With PGW Interaction) Call Flow

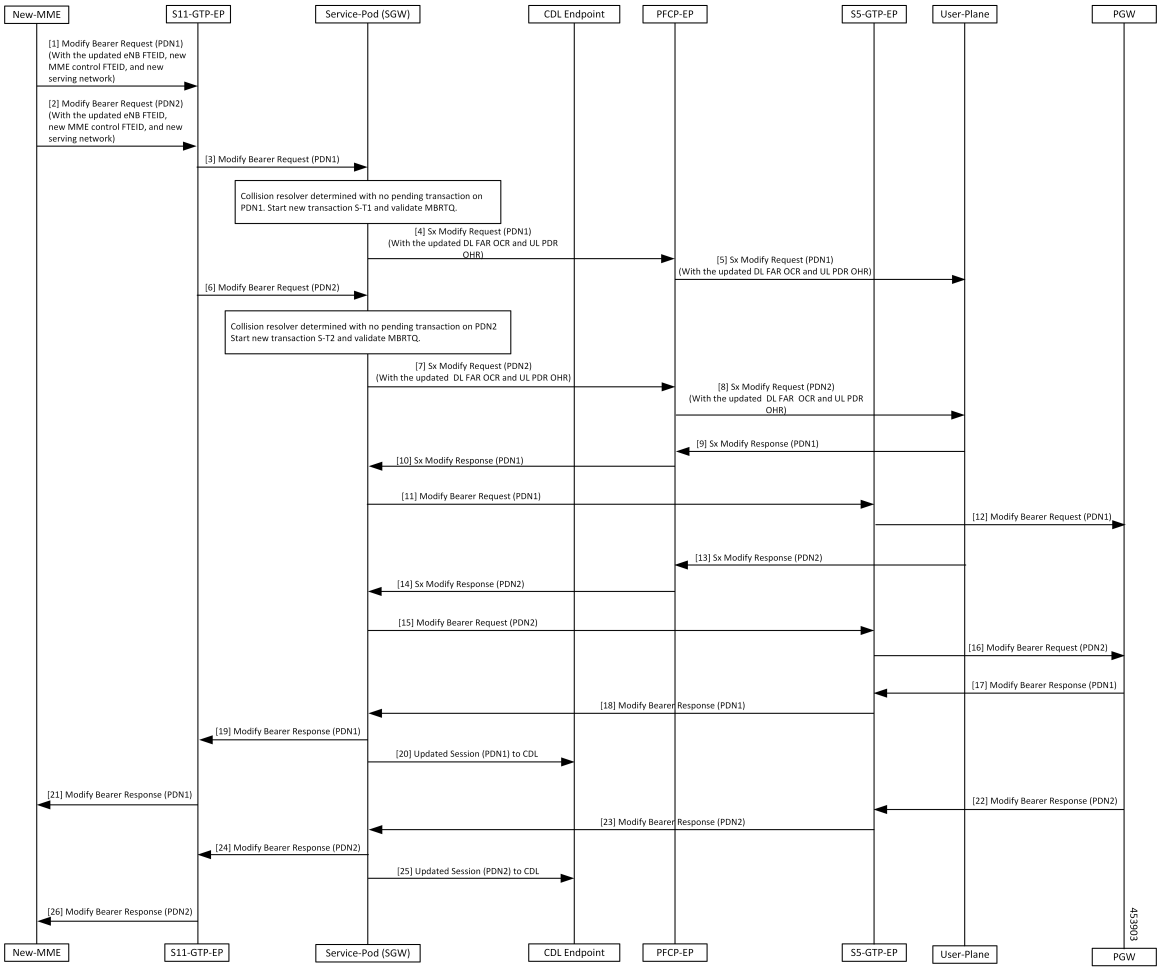


Table 157: Inter MME Handover with Multi-PDN Handling (With PGW Interaction) Call Flow Description

Step	Description
1	The New-MME sends the Modify Bearer Request for (PDN1) with the updated eNodeB FTEID, new MME control FTEID, and new serving network to the S11-GTP-EP.
2	The New-MME sends the Modify Bearer Request for (PDN2) with the updated eNodeB FTEID, new MME control FTEID, and new serving network to the S11-GTP-EP.
3	The S11-GTP-EP sends the Modify Bearer Request (PDN1) to the cnSGW-C.
4	The cnSGW-C sends the Sx Modify Request (PDN1) with the updated DL FAR OCR and UL PDR OHR to the PFCP-EP.
5	The PFCP-EP forwards the Sx Modify Request for (PDN1) to the User-Plane.

Step	Description
6	The S11-GTP-EP sends the Modify Bearer Request (PDN2).
7	The Service-Pod (SGW) sends Sx Modify Request (PDN2) with updated DL FAR OCR and UL PDR OHR to the PFCP-EP.
8	The PFCP-EP forwards the Sx Modify Request (PDN2) to User-Plane.
9	The User-Plane sends the Sx Modify Response (PDN1) to the PFCP-EP.
10	The PFPC-EP forwards the Sx Modify Response (PDN1) to the Service-Pod (SGW).
11	The Service-Pod (SGW) sends the Modify Bearer Request (PDN1) with updated serving network information to the S5-GTP-EP.
12	The S5-GTP-EP forwards the Modify Bearer Request (PDN1) to the PGW.
13	The User-Plane sends the Sx Modify Response (PDN2) to the PFCP-EP.
14	The PFCP-EP forwards the Sx Modify Response (PDN2) to the Service-Pod (SGW).
15	The Service-Pod (SGW) sends the Modify Bearer Request (PDN2) with the updated serving network information to S5-GTP-EP.
16	The S5-GTP-EP sends the Modify Bearer Request (PDN2) to the PGW.
17	The PGW sends the Modify Bearer Response (PDN1) to the S5-GTP-EP.
18	The S5-GPT-EP forwards the Modify Bearer Response (PDN1) to the Service-Pod (SGW).
19	The Service-Pod (SGW) forwards the Modify Bearer Response (PDN1) to the S11-GTP-EP.
20	The Service-Pod (SGW) updates the PDN1 session sent to the CDL Endpoint.
21	The S11-GTP-EP sends the Modify Bearer Response (PDN1) to the New-MME.
22	The PGW sends the Modify Bearer Response for (PDN2) to the S5-GTP-EP.
23	The S5-GPT-EP forwards the Modify Bearer Response (PDN2) to the Service-Pod (SGW).
24	The Service-Pod (SGW) forwards the Modify Bearer Response (PDN2) to the S11-GTP-EP.
25	The Service-Pod (SGW) marks Inter-MME as completed (PDN2) and updates PDN2 session sent to the CDL Endpoint.
26	The S11-GTP-EP sends the Modify Bearer Response (PDN2) to New-MME.

Multi PDN Call X2 Handover SGW Relocation to cnSGW-C Call Flow

This section describes the Multi PDN Call X2 Handover SGW Relocation to cnSGW-C call flow.

Figure 82: Multi PDN Call X2 Handover SGW Relocation to cnSGW-C Call Flow

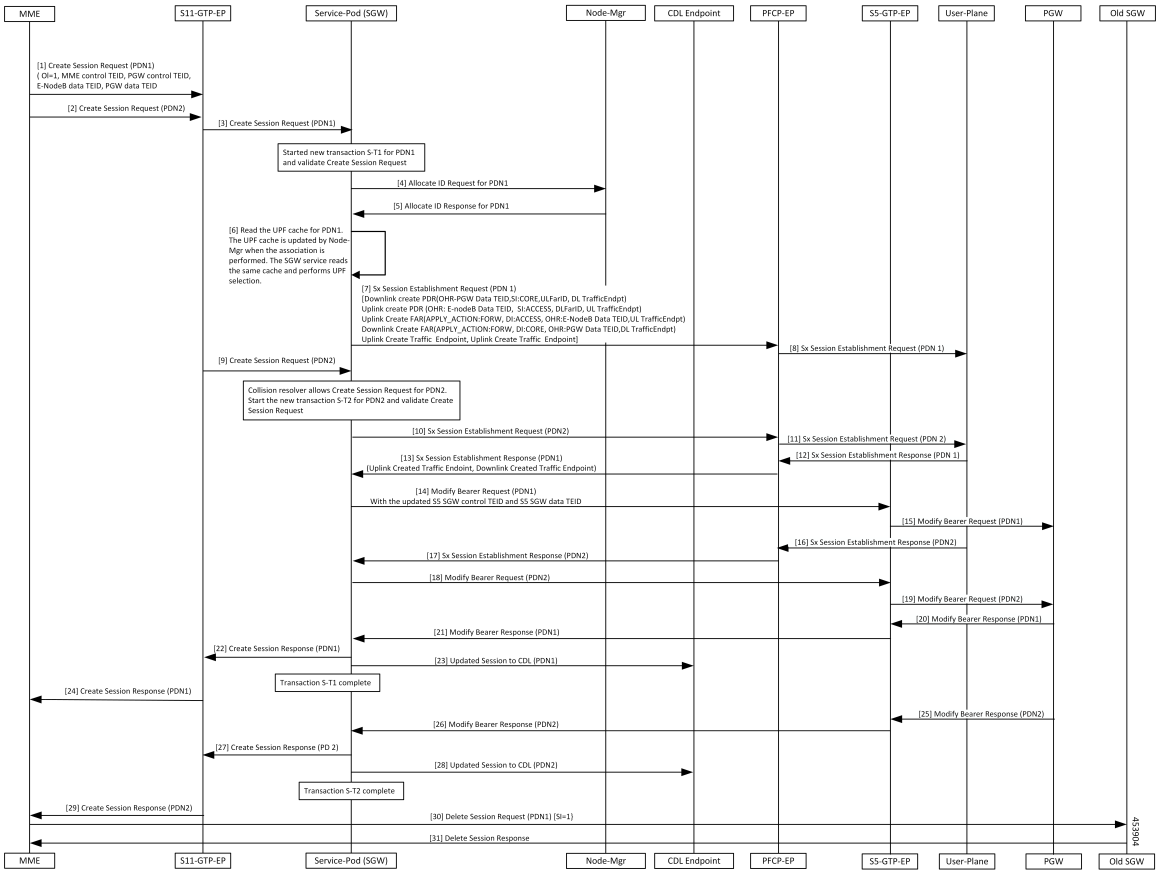


Table 158: Multi PDN call X2 Handover SGW Relocation to cnSGW-C Call Flow Description

Step	Description
1	The MME sends the Create Session Request for (PDN1) with (OI = 1, MME Control TEID, PGW Control TEID, eNodeB Data TEID, PGW Data TEID) to the S11-GTP-EP.
2	The MME sends the Create Session Request (PDN2) to the S11-GTP-EP.
3	The S11-GTP-EP forwards the Create Session Request (PDN1) to the Service-Pod (SGW).
4	The Service-Pod (SGW) requests for ID allocation (PDN1) to the Node-Mgr.
5	The Node-Mgr responds with the Allocate Id Response (PDN1).
6	The Service-Pod (SGW) performs the UPF selection.
7	The Service-Pod (SGW) sends the Sx Session Establishment Request (PDN1) to the PFCP-EP.
8	The PFCP-EP forwards the Sx Session Establishment Request (PDN1) to the User-Plane.
9	The S11-GTP-EP sends the Create Session Request (PDN2) to the Service-Pod (SGW).

Step	Description
10	The Service-Pod (SGW) sends the Sx Session Establishment Request (PDN2) to PFCP-EP.
11	The PFCP-EP forwards the Sx Session Establishment Request (PDN2) to the User-Plane.
12	The User-Plane sends the Sx Session Establishment Response (PDN1) to the PFCP-EP.
13	The PFCP-EP sends the Sx Session Establishment Response (PDN1) to the Service-Pod (SGW).
14	The Service-Pod (SGW) sends the Modify Bearer Request (PDN1) with the updated S5 SGW Control TEID and S5 SGW Data TEID to S5-GTP-EP.
15	The S5-GTP-EP forwards the Modify Bearer Request (PDN1) to the PGW.
16	The User-Plane sends the Sx Session Establishment Response (PDN2) to the PFCP-EP.
17	The PFCP-EP forwards the Sx Session Establishment Response (PDN2) to the Service-Pod (SGW).
18	The Service-Pod (SGW) sends the Modify Bearer Request (PDN2) with the updated S5 SGW Control TEID and S5 SGW Data TEID to the S5-GTP-EP.
19	The S5-GTP-EP forwards the Modify Bearer Request (PDN2) to the PGW.
20	The PGW sends the Modify Bearer Response (PDN1) to the S5-GTP-EP.
21	The S5-GTP-EP forwards the Modify Bearer Response (PDN1) to the Service-Pod (SGW).
22	The Service-Pod (SGW) sends the Create Session Response (PDN1) to the S11-GTP-EP.
23	The Service-Pod (SGW) updates the PDN1 session to the CDL Endpoint.
24	The S11-GTP-EP forwards the Create Session Response (PDN1) to the MME.
25	The PGW sends the Modify Bearer Response (PDN2) to the S5-GTP-EP.
26	The S5-GTP-EP forwards the Modify Bearer Response (PDN2) to the Service-Pod (SGW).
27	The Service-Pod (SGW) sends the Create Session Response (PDN2) to the S11-GTP-EP.
28	The Service-Pod (SGW) updates the PDN2 session to the CDL Endpoint.
29	The S11-GTP-EP forwards the Create Session Response (PDN2) to the MME.
30	The MME sends the Delete Session Request (PDN1) [SI=1] to the Old SGW.
31	The Old SGW responds with the Delete Session Response to the MME.

Multi-PDN S1 Handover SGW Relocation to Service-Pod (SGW) Call Fow

This section describes the Multi-PDN S1 Handover SGW Relocation to Service-Pod (SGW) call flow.

Figure 83: Multi-PDN S1 Handover SGW Relocation to Service-Pod (SGW) Call Flow

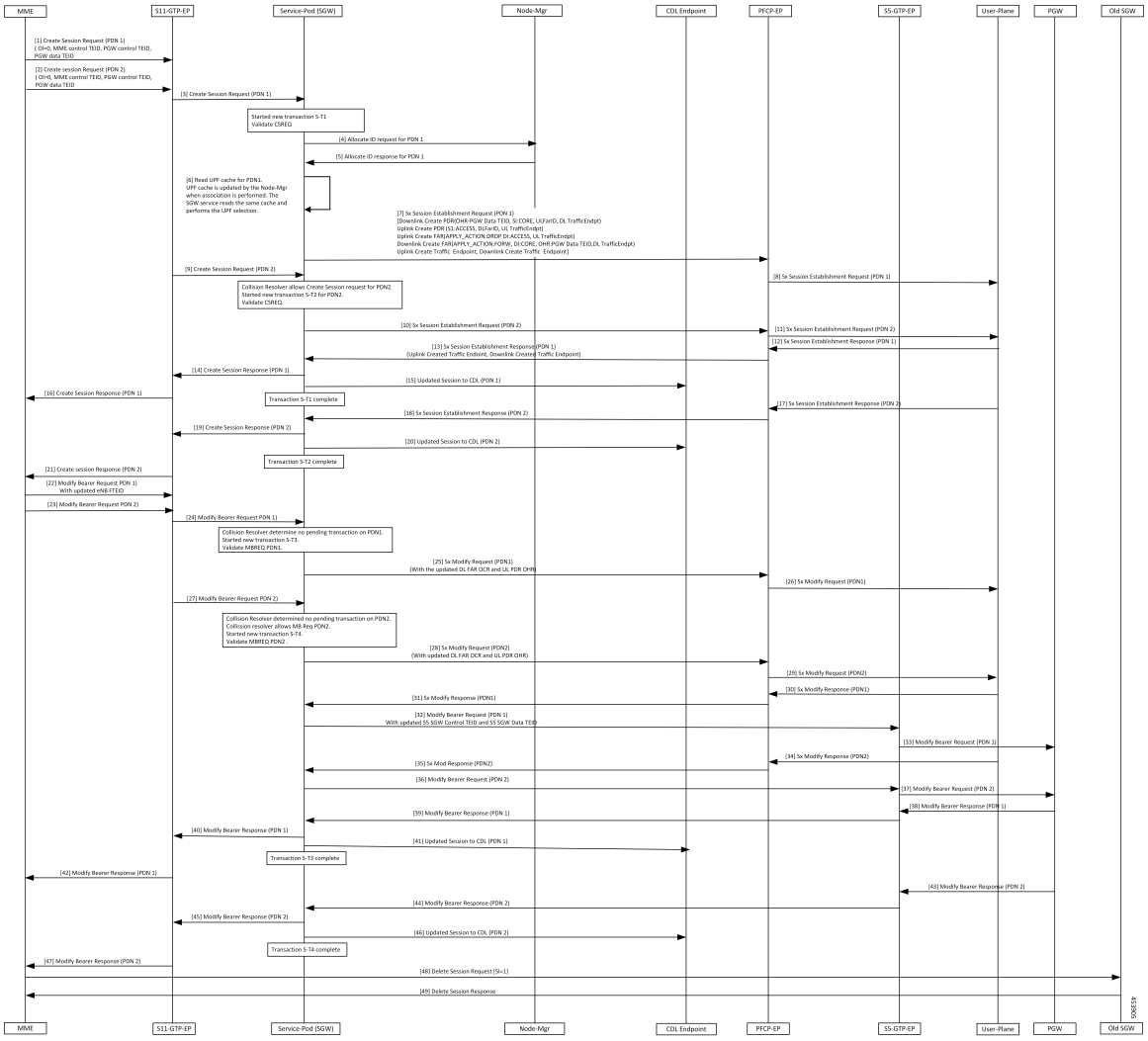


Table 159: Multi-PDN S1 Handover SGW Relocation to Service-Pod (SGW) Call Flow Description

Step	Description
1	The MME sends a Create Session Request (PDN1) with (OI = 0, MME Control TEID, PGW Control TEID, and PGW Data TEID) to the S11-GTP-EP.
2	The MME sends the Create Session Request (PDN2) with (OI = 0, MME Control TEID, PGW Control TEID, and PGW Data TEID) to the S11-GTP-EP.
3	The S11-GTP-EP sends the Create Session Request (PDN1) to the Service-Pod (SGW).
4	The Service-Pod (SGW) requests ID allocation (PDN1) to the Node-Mgr.
5	The Node-Mgr responds with the Allocate ID Response (PDN1) to the Service-Pod (SGW).
6	The Service-Pod (SGW) performs the UPF selection.

Step	Description
7	The Service-Pod (SGW) triggers the Sx Session Establishment Request (PDN1) to the PFCP-EP.
8	The PFCP-EP forwards a Sx Session Establishment Request (PDN1) to the User-Plane.
9	The Service-Pod (SGW) receives the Create Session Request (PDN2) from the S11-GTP-EP.
10	The Service-Pod (SGW) sends a Sx Session Establishment Request (PDN2) to the PFCP-EP.
11	The PFCP-EP forwards the Sx Session Establishment Request (PDN2) to the User-Plane.
12	The User-Plane responds with the Sx Session Establishment Response (PDN1) to the PFCP-EP.
13	The PFCP-EP forwards the Sx Session Establishment Response (PDN1) to the Service-Pod (SGW).
14	The Service-Pod (SGW) sends the Create Session Response (PDN1) to the S11-GTP-EP.
15	The Service-Pod (SGW) updates the PDN1 session to the CDL Endpoint.
16	The S11-GTP-EP forwards the Create Session Response (PDN1) to the MME.
17	The User-Plane responds with the Sx Session Establishment Response (PDN2) to the PFCP-EP.
18	The PFCP-EP forwards the Sx Session Establishment Response (PDN2) to the Service-Pod (SGW).
19	The Service-Pod (SGW) sends the Create Session Response (PDN2) to the S11-GTP-EP.
20	The Service-Pod (SGW) updates the PDN2 session to the CDL Endpoint.
21	The S11-GTP-EP forwards the Create Session Response (PDN2) to the MME.
22	The MME sends a Modify Bearer Request (PDN1) to the S11-GTP-EP.
23	The MME sends the Modify Bearer Request (PDN2) to the S11-GTP-EP.
24	The S11-GTP-EP forwards the Modify Bearer Request (PDN1) to the Service-Pod (SGW).
25	The Service-Pod (SGW) sends a Sx Modify Request (PDN1) to the PFCP-EP.
26	The PFCP-EP forwards the Sx Modify Request (PDN1) to the User-Plane.
27	The S11-GTP-EP sends the Modify Bearer Request (PDN2) to the Service-Pod (SGW).
28	The Service-Pod (SGW) sends a Sx Modify Request (PDN2) with the updated DL FAR OCR and UL PDR OHR to the PFCP-EP.
29	The PFCP-EP forwards the Sx Modify Request (PDN2) to the User-Plane.
30	The User-Plane responds with the Sx Modify Response (PDN1) to the PFCP-EP.
31	The PFCP-EP forwards the Sx Modify Response (PDN1) to the Service-Pod (SGW).
32	The Service-Pod (SGW) sends the Modify Bearer Request (PDN1) with the updated S5 SGW Control TEID and S5 SGW Data TEID to the S5-GTP-EP.
33	The S5-GTP-EP forwards the Modify Bearer Request (PDN1) to the PGW.

Step	Description
34	The User-Plane sends the Sx Modify Response (PDN2) to the PFCP-EP.
35	The PFCP-EP forwards the Sx Modify Response (PDN2) to the Service-Pod (SGW).
36	The Service-Pod (SGW) sends the Modify Bearer Request (PDN2) to the S5-GTP-EP.
37	The S5-GTP-EP forwards the Modify Bearer Request (PDN2) to the PGW.
38	The PGW responds with the Modify Bearer Response (PDN1) to the S5-GTP-EP.
39	The S5-GTP-EP forwards the Modify Bearer Response (PDN1) to the Service-Pod (SGW).
40	The Service-Pod (SGW) sends the Modify Bearer Response (PDN1) to the S11-GTP-EP.
41	The Service-Pod (SGW) updates the PDN1 session to the CDL Endpoint.
42	The S11-GTP-EP forwards the Modify Bearer Response (PDN1) to the MME.
43	The PGW responds with the Modify Bearer Response (PDN2) to the S11-GTP-EP.
44	The S5-GTP-EP forwards the Modify Bearer Response (PDN2) to the Service-Pod (SGW).
45	The Service-Pod (SGW) sends the Modify Bearer Response (PDN2) to the S11-GTP-EP.
46	The Service-Pod (SGW) updates the PDN2 session to the CDL Endpoint.
47	The S11-GTP-EP forwards the Modify Bearer Response (PDN2) to the MME.
48	The MME triggers the Delete Session Request [SI=1] to Old SGW.
49	The Old SGW deletes the session and responds with the Delete Session Response.

Multiple CBR for Same PDN Call Flow

This section describes the Multiple CBR for Same PDN call flow.

Figure 84: Multiple CBR for Same PDN Call Flow

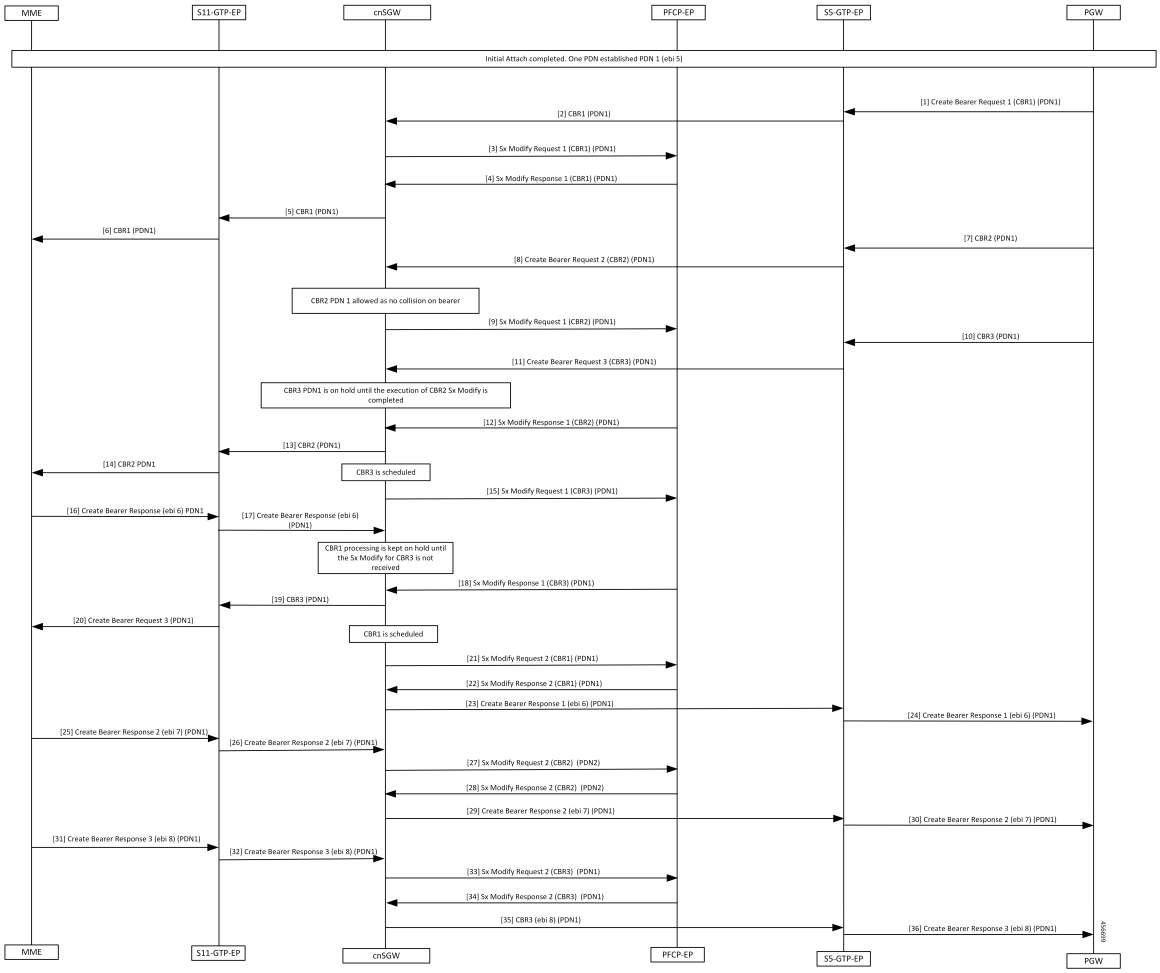


Table 160: Multiple CBR for Same PDN Call Flow Description

Step	Description
1	Initial Attach completed. One PDN has established PDN1 (ebi 5). The PGW sends the Create Bearer Request 1 for PDN1 to the S5-GTP-EP.
2	The S5-GTP-EP forwards the Create Bearer Request 1 for PDN1 to the cnSGW.
3	The cnSGW sends the Sx Modify Request 1 for PDN1 to the PFCP-EP.
4	The cnSGW receives the Sx Modify Response 1 for PDN1 from the PFCP-EP.
5	The cnSGW sends the Create Bearer Request 1 for PDN1 to the S11-GTP-EP.
6	The S11-GTP-EP sends the Create Bearer Request 1 for PDN1 to the MME.
7	The PGW sends the Create Bearer Request 2 to the S5-GTP-EP.

Step	Description
8	The S5-GTP-EP sends the Create Bearer Request 2 to the cnSGW.
9	The cnSGW sends the Sx Modify Request 1 (Create Bearer Request 2) for PDN1 to the PFCP-EP.
10	The PGW sends the Create Bearer Request 3 for PDN1 to the S5-GTP-EP.
11	The S5-GTP-EP sends the Create Bearer Request 3 for PDN1 to the cnSGW
12	The PFCP-EP sends the Sx Modify Response 1 (CBR2) for PDN1 to the cnSGW.
13	The cnSGW sends the Create Bearer Request 2 for PDN1 to the S11-GTP-EP.
14	The S11-GTP-EP sends the Create Bearer Request 2 for PDN1 to the MME.
15	The cnSGW sends the Sx Modify Request 1(CBR3) for PDN1 to the PFPC-EP.
16	The MME sends the Create Bearer Response 1 (ebi6) for PDN1 to the S11-GTP-EP.
17	The S11-GTP-EP sends the Create Bearer Response 1 (ebi6) for PDN1 to the cnSGW.
18	The cnSGW receives the Sx Modify Response 1 (CBR3) for PDN1 to the PFPC-EP.
19, 20	The cnSGW sends the Create Bearer Request 3 to the S11-GTP-EP.
20	The S11-GTP-EP sends the Create Bearer Request 3 to the MME.
21	The cnSGW sends the Sx Modify Request 2(CBR1) for PDN1 to the PFPC-EP.
22	The cnSGW receives the Sx Modify Response 2 (CBR1) for PDN1 to the PFPC-EP.
23	The cnSGW sends the Create Bearer Response 1 (ebi 6) for PDN1 to the S5-GTP-EP.
24	The S5-GTP-EP sends the Create Bearer Response 1 (ebi 6) for PDN1 to the PGW.
25	The MME sends the Create Bearer Response 2 (ebi 7) for PDN1 sent from MME to cnSGW.
26	The S11-GTP-EP sends the Create Bearer Response 2 (ebi 7) for PDN1 sent from MME to the cnSGW.
27	The cnSGW sends the Sx Modify Request 2 (CBR2) for PDN1 to PFCP-EP.
28	The PFCP-EP sends the Sx Modify Response 2 (CBR2) for PDN1 to the cnSGW.
29	The cnSGW sends the Create Bearer Response 2 (ebi 7) for PDN1 to the S5-GTP-EP.
30	The S5-GTP-EP sends the Create Bearer Response 2 (ebi 7) for PDN1 to the PGW.
31	The MME sends the Create Bearer Response 3 (ebi 8) for PDN1 to the S11-GTP-EP.
32	The S11-GTP-EP sends the Create Bearer Response 3 (ebi 8) for PDN1 to the cnSGW.
33	The cnSGW sends the Sx Modify Request 2 (CBR3) for PDN1 to the PFPC-EP.
34	The PFCP-EP send the Sx Modify Response 2 (CBR3) for PDN1 to the cnSGW.

Step	Description
35	The cnSGW sends the Create Bearer Response 3 (ebi 8) for PDN1 to the S5-GTP-EP.
36	The S5-GTP-EP sends the Create Bearer Response 3 (ebi 8) for PDN1 to the PGW.

Collision Resolver Discard Handling Call Flow

This section describes the Collision Resolver Discard Handling call flow.

Figure 85: Collision Resolver Discard Handling Call Flow

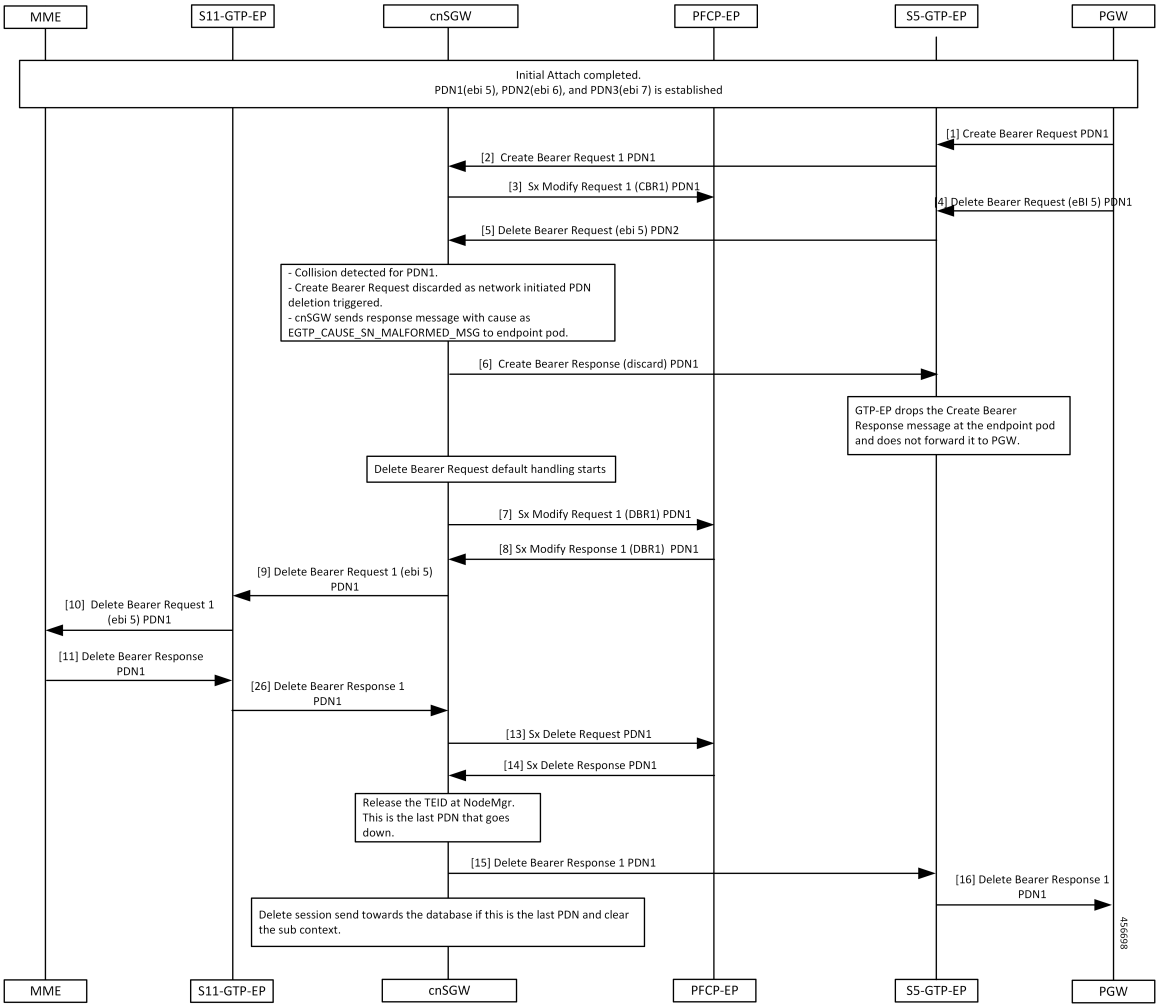


Table 161: Collision Resolver Discard Handling Call Flow Description

Step	Description
1	Initial Attach procedure is completed. The PDN1(ebi 5), PDN2(ebi 6), and PDN3(ebi 7) is established. The PGW sends the Create Bearer Request for PDN1 to the S5-GTP-EP.

Step	Description
2	The PFCP-EP sends the Create Bearer Request 1 for PDN1 to the cnSGW.
3	The cnSGW sends the Sx Modify Request 1 for PDN1 to the PFCP-EP.
4, 5	The PGW sends the Delete Bearer Request (ebi5) for PDN1 to the cnSGW.
6	Collision detected for PDN1. The Create Bearer Request is discarded as network initiated PDN deletion request is triggered. The cnSGW sends a response message with the cause as EGTP_CAUSE_SN_MALFORMED_MSG to the endpoint pod. The cnSGW sends Create Bearer Response (discard 1) for PDN1 to the S5-GTP-EP.
7	The cnSGW sends the Sx Modify Request 1 (DBR1) for PDN1 to the PFPC-EP.
8	The cnSGW receives the Sx Modify Response 1 (DBR1) for PDN1 from the PFPC-EP.
9	The cnSGW sends the Delete Bearer Request 1 to the S11-GTP-EP.
10	The S11-GTP-EP sends the Delete Bearer Request 1 to the MME.
11	The MME sends the Delete Bearer Response 1 to the S11-GTP-EP.
12	The S11-GTP-EP sends the Delete Bearer Response 1 to the cnSGW.
13	The cnSGW sends the Sx Delete Request for PDN1 to the PFCP-EP.
14	The PFPC-EP sends the Sx Delete Response for PDN1 to the cnSGW.
15	The cnSGW sends the Delete Bearer Response 1 for PDN1 to the PGW.
16	The S5-GTP-EP sends the Delete Bearer Response for PDN1 to the PGW.

Suspend Handling Call Flow

This section describes the Suspend Handling call flow.

Figure 86: Suspend Handling Call Flow

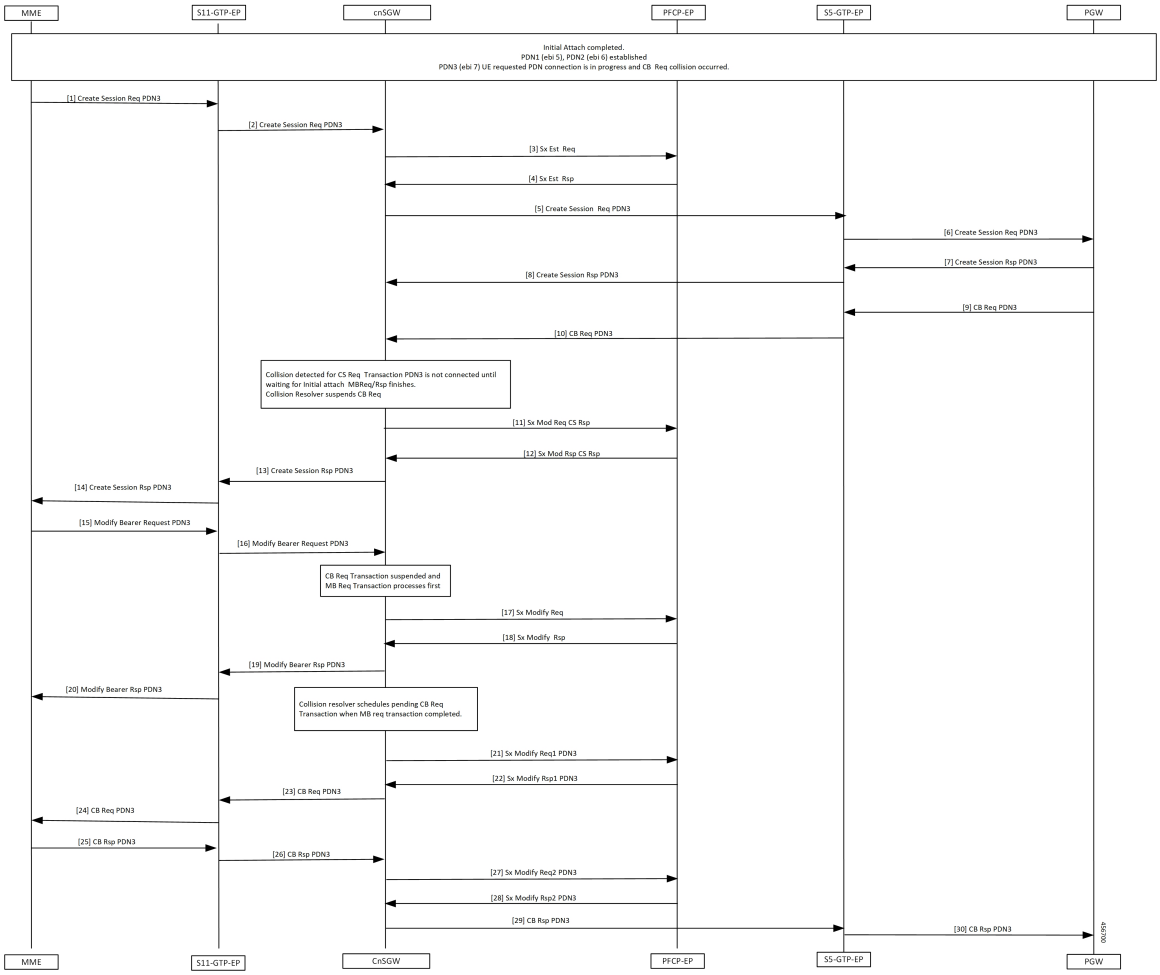


Table 162: Suspend Handling Call Flow Description

Step	Description
1	Initial Attach process is completed. The PDN1 (ebi 5) and PDN2 (ebi 6) is established. PDN3 (ebi 7) UE requested that PDN connection is in progress. The Create Bearer Request collision occurred. The MME sends Create Session Req for PDN3 to S11-GTP-EP.
2	The S11-GTP-EP sends the Create Session Request for PDN3 to the cnSGW.
3	The cnSGW sends the Sx Establishment Request to the PFCP-EP.
4	The PFCP-EP sends the Sx Establishment Response to the cnSGW.
5	The cnSGW sends the Create Session Request for PDN3 to the PGW.

Step	Description
6	The S5-GTP-EP sends the Create Session Request for PDN3 to the PGW.
7	The PGW sends the Create Session Rsppnse for PDN3 to the S5-GTP-EP.
8	The S5-GTP-EP sends the Create Session Response for PDN3 to the cnSGW.
9	The PGW sends the Create Bearer Request for PDN3 to the S5-GTP-EP.
10	The S5-GTP-EP sends the Create Bearer Request for PDN3 to the cnSGW.
11	Collision that is detected for Create Session Request transaction for PDN3 gets connected when waiting for the initial attach Modify Bearer Request or Response is complete. Collision Resolver suspends Create Bearer Request. The cnSGW sends Sx Modify Request and Create Session Request to the PFCP-EP.
12	The PFCP-EP sends the Sx Modify Response and Create Session Response to cnSGW.
13	The cnSGW sends the Create Session Response for PDN3 sent to the S11-GTP-EP.
14	The S11-GTP-EP sends the Create Session Response for PDN3 sent to the MME.
15	The MME sends the Modify Bearer Request for PDN3 to the S11-GTP-EP.
16	The S11-GTP-EP sends the Modify Bearer Request for PDN3 to the cnSGW.
17	The cnSGW sends the Modify Bearer Request for PDN3 to the PFCP-EP.
18	The PFCP-EP sends the Sx Modify Response to the cnSGW.
19	The cnSGW sends the Modify Bearer Response for PDN3 to the S11-GTP-EP.
20	The S11-GTP-EP sends the Modify Bearer Response for PDN3 to the MME.
21	The cnSGW sends Sx Modify Request 1 for PDN3 to the PFCP-EP.
22	The PFPC-EP sends the Sx Modify Response 1 for PDN3 to the PFPC-EP.
23	The cnSGW sends the Create Bearer Request for PDN3 sent to the S11-GTP-EP.
24	The S11-GTP-EP sends the Create Bearer Request for PDN3 sent to the MME.
25	The MME sends the Create Bearer Response for PDN3 to the S11-GTP-EP.
26	The S11-GTP-EP sends the Create Bearer Response for PDN3 to the cnSGW.
27	The cnSGW sends Sx Modify Request 2 for PDN3 to the PFCP-EP.
28	The PFCP-EP sends the Sx Modify Response 2 for PDN3 to the cnSGW.
29	The cnSGW sends the Create Bearer Response for PDN3 to the PGW.
30	The S5-GTP-EP sends the Create Bearer Response for PDN3 to the PGW.

Abort Handling of Low-Priority Procedure Call Flow

This section describes the Abort Handling of Low-Priority Procedure call flow.

Figure 87: Abort Handling of Low-Priority Procedure Call Flow

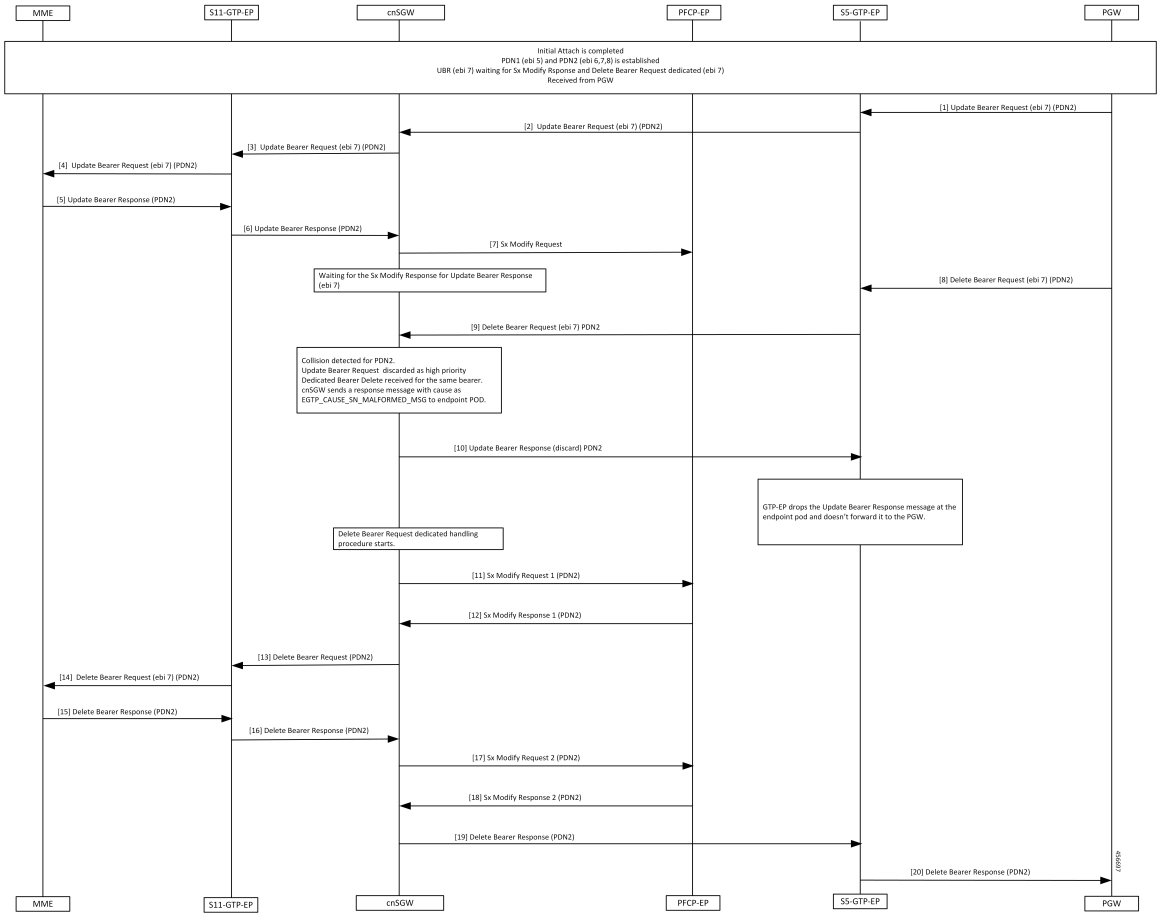


Table 163: Abort Handling of Low-Priority Procedure Call Flow Description

Step	Description
1	Initial Attach procedure is completed. The PDN1(ebi 5) and PDN2(ebi 6,7,8) is established. The Update Bearer Request (ebi 7) is waiting for the Sx Modify Response and Dedicated Bearer Request dedicated (ebi 7) received from the PGW. The PGW sends an Update Bearer Request (eBi 7) PDN2 to the S5-GTP-EP.
2	The PFCP-EP sends the Update Bearer Request (eBi 7) PDN2 to the cnSGW.
3	The cnSGW sends the Update Bearer Request (eBi 7) PDN2 to the S11-GTP-EP.
4	The S5-GTP-EP forwards the Update Bearer Request (eBi 7) PDN2 to the MME.

Step	Description
5	The MME sends an Update Bearer Response to the S11-GTP-EP.
6	The S11-GTP-EP sends a Update Bearer Response to the cnSGW.
7	The cnSGW sends Sx Modify Request to the PFCP-EP.
8	The PGW sends the Delete Bearer Request (eBi 7) for PDN2 to the S5-GTP-EP.
9	The S5-GTP-EP forwards Delete Bearer Request (eBi 7) for PDN2 to the cnSGW.
10	Collision is detected for PDN2. The Update Bearer Request is discarded as high priority-dedicated bearer delete received for the same bearer. The cnSGW sends a response message with cause as EGTP_CAUSE_SN_MALFORMED_MSG to endpoint POD. The cnSGW sends Update Bearer Response (Discard) for PDN2 to the S5-GTP-EP.
11	The GTP-EP drops the Update Bearer Response message at endpoint pod and not forwarded to the PGW. The cnSGW sends the Sx Modify Request 1 for PDN2 to the PFCP-EP.
12	The PFCP-EP sends the Sx Modify Response 1 for PDN2 to the cnSGW.
13	The cnSGW sends the Delete Bearer Request for PDN2 to the S11-GTP-EP.
14	The S11-GTP-EP sends the Delete Bearer Request (ebi 7) for PDN2 to the MME.
15	The MME sends the Delete Bearer Response for PDN2 to the S11-GTP-EP.
16	The S11-GTP-EP sends the Delete Bearer Response for PDN2 to the cnSGW.
17	The cnSGW sends the Sx Modify Request for PDN2 to the PFCP-EP.
18	The PFCP-EP sends Sx Modify Response for PDN2 to the cnSGW.
19	The cnSGW sends the Delete Bearer Response for PDN2 to the S5-GTP-EP.
20	The S5-GTP-EP sends the Delete Bearer Response for PDN2 to the PGW.

Double Delete Optimization Call Flow

This section describes the Double Delete Optimization call flow.

Figure 88: Double Delete Optimization Call Flow

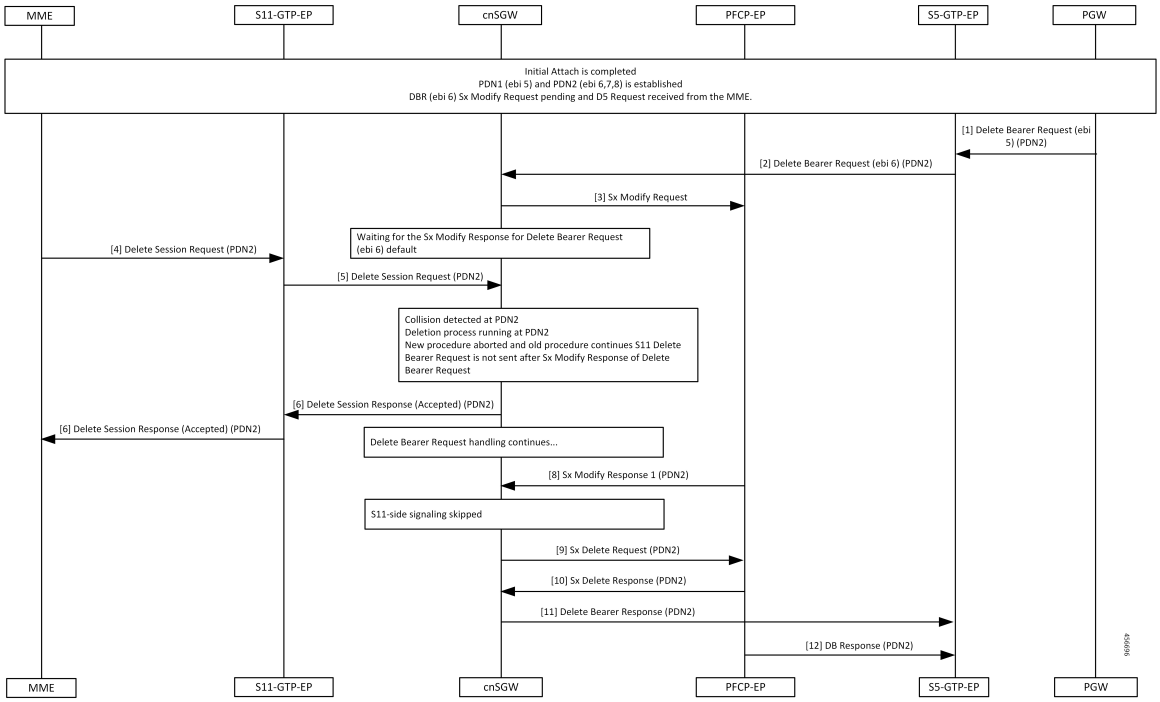
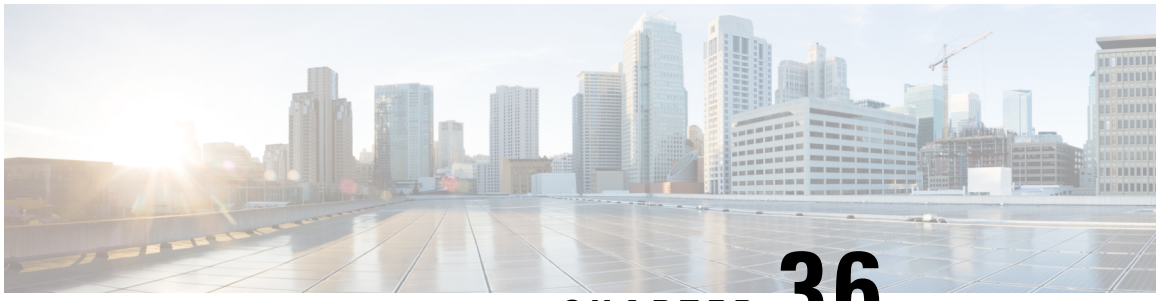


Table 164: Double Delete Optimization Call Flow Description

Step	Description
1	Initial Attach procedure is completed. The PDN1(ebi 5) and PDN2(ebi 6,7,8) is established. The Delete Bearer Request (ebi 6) and Sx Modify Request is pending and the Delete Session Request is received from the MME. The PGW sends the Delete Bearer Request (ebi 6) for PDN2 to the S5-GTP-EP.
2	The S5-GTP-EP forwards the Delete Bearer Request (ebi 6) for PDN2 to the cnSGW.
3	The cnSGW sends the Sx Modify Request to the PFCP-EP.
4	Waits for the Sx Modify Response for the Delete Bearer Request (ebi 6) default. The MME sends the Delete Session Request for PDN2 to the S11-GTP-EP.
5	The S11-GTP-EP forwards the Delete Session Request for PDN2 to the cnSGW.

Step	Description
6	Collision detected at PDN2. Deletion process running at PDN2. New procedure aborted and old procedure continue. The S11 Delete Bearer Request isn't send after an Sx Modify Response of the Delete Bearer Request. The cnSGW sends Delete Session Response (Accepted) for PDN2 to the S11-GTP-EP.
7	The S11-GTP-EP forwards the Delete Session Response (Accepted) for PDN2 to the MME.
8	While the Delete Bearer Request handling continues, the PFCP-EP sends Sx Modify Response 1 for PDN2 to the cnSGW.
9	The cnSGW sends the Sx Delete Request for PDN2 to the PFCP-EP.
10	The PFCP-EP sends the Sx Delete Response for PDN2 to the cnSGW.
11	The cnSGW sends Delete Bearer Response for PDN2 to the S5-GTP-EP.
12	The S5-GTP-EP forwards the Delete Bearer Response for PDN2 to the PGW.



CHAPTER 36

Modify and Delete Bearer Command Support

- Feature Summary and Revision History, on page 383
- Feature Description, on page 383
- How it Works, on page 384

Feature Summary and Revision History

Summary Data

Table 165: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled – Always-on
Related Documentation	Not Applicable

Revision History

Table 166: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

cnSGW-C supports Modify Bearer Command (MBC) and Delete Bearer Command (DBC).

This feature is supported on the following pods—SGW-service, GTP-EP, and UDP-Proxy.

The SGW-service pod is responsible for handling the following:

- The MBC and DBC
- The MBC triggered Update Bearer Request
- The DBC triggered Delete Bearer Response

The GTPC-EP pod is responsible for sending the following:

- Modify Bearer Command Failure Indication (MBCFI) and Delete Bearer Command Failure Indication (DBCFI) if no response is received.
- MBCFI and DBCFI (success) on receiving Update Bearer Request and Delete Bearer Request respectively.
- Update Bearer Response and Delete Bearer Response back to PGW on receiving the respective message from the SGW-service pod.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

MBC Failure Handling Call Flow

This section describes the MBC Failure Handling call flow.

Figure 89: MBC Failure Handling Call Flow

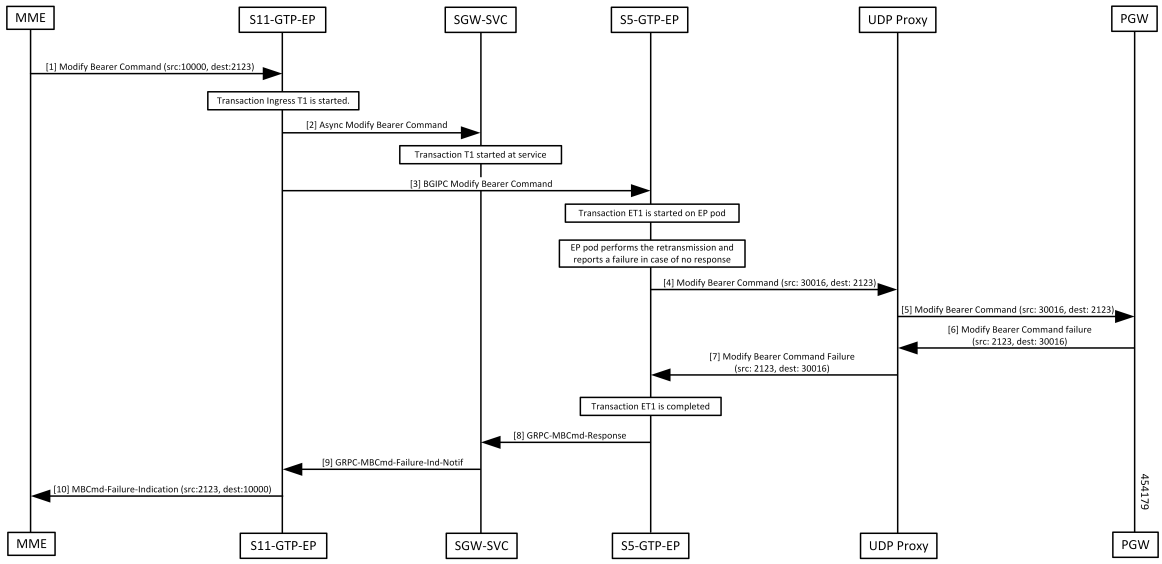


Table 167: MBC Failure Handling Call Flow Description

Step	Description
1	The MME sends the Modify Bearer Command to the S11 GTPC-EP pod.
2	The GTPC-EP pod sends the ASYNC Modify Bearer Command to the SGW-SVC pod.
3	The SGW-SVC pod forwards Modify Bearer Command to PGW. Save MBC_info in PDN for the response.
4	The GTPC-EP pod performs retransmission. If there is no response, the pod sends Modify Bearer Command Failure Indication (MBCFI) to SGW-SVC pod.
5	The UDP Proxy sends the Modify Bearer Command request to PGW.
6	The MBCFI is received on the S5 GTPC-EP pod and is forwarded to SGW-SVC pod.
7	The UDP Proxy sends the Modify Bearer Command failure details to the S5-GTP-EP.
8	The S5-GTP-EP sends the GRPC Modify Bearer Command Response to the SGW-SVC.
9	The SGW-SVC sends the GRPC Modify Bearer Command failure notification to the S11-GTP-EP.
10	The SGW-SVC pod processes MBCFI and forwards the response to MME with saved MBC_Info.

MBC Success Handling Call Flow

This section describes the MBC Success Handling call flow.

Figure 90: MBC Success Handling Call Flow

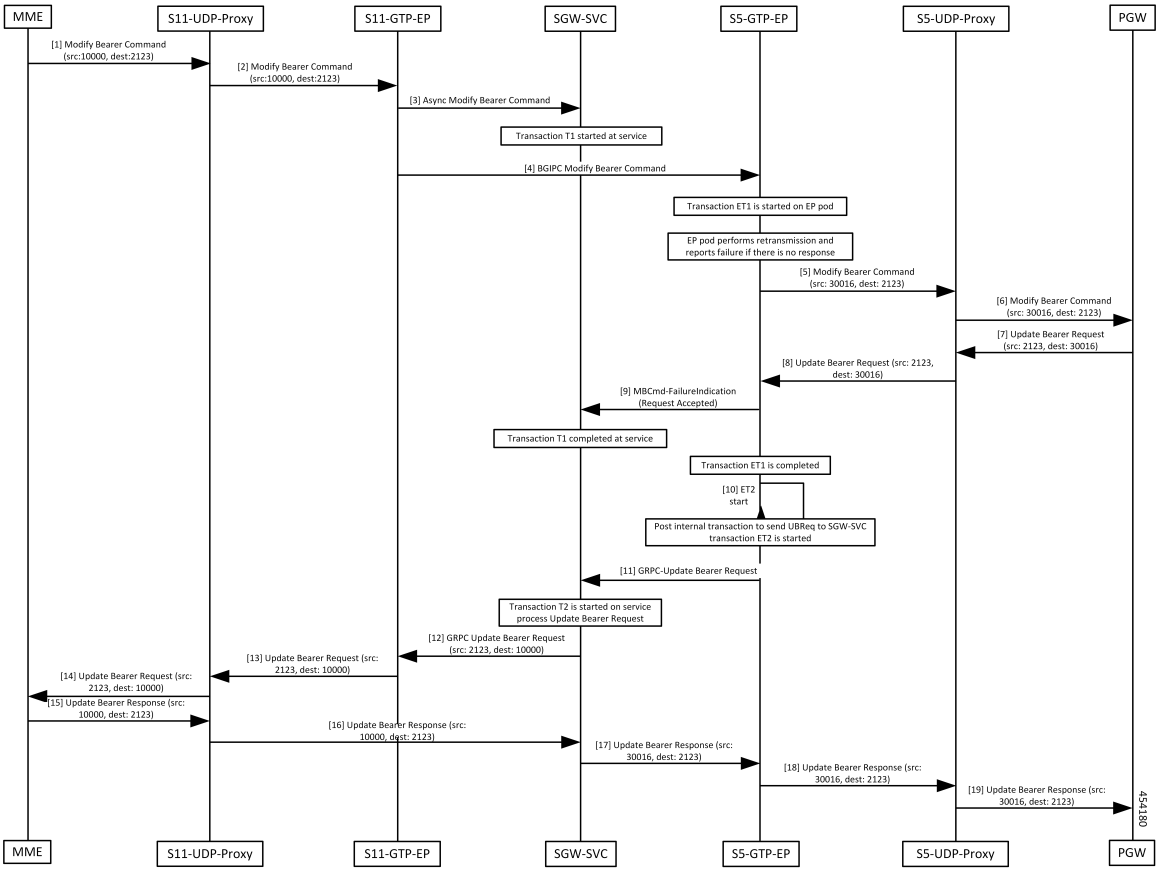


Table 168: MBC Success Handling Call Flow Description

Step	Description
1	The MME sends the Modify Bearer Command to the S11-UDP-Proxy.
2	The S11-UDP-Proxy forwards the Modify Bearer Command to the S11-GTP-EP pod.
3	The S11-GTP-EP pod sends the ASYNC Modify Bearer Command to the SGW-SVC pod. The SGW-SVC pod forwards the Modify Bearer Command to the PGW. Save MBC_info in PDN for response.
4	The S11-GTP-EP pod performs the retransmission. If there is no response, the pod sends the Modify Bearer Command Failure Indication (MBCFI) to the S5-GTP-EP pod.
5	The S5-GTP-EP sends the Modify Bearer Command to the S5-UDP-Proxy.
6	The S5-UDP-Proxy forwards the Modify Bearer Command to the PGW.
7	The PGW sends the Update Bearer Request to the S5-UDP-Proxy.
8	The S5-UDP-Proxy sends the Update Bearer Request (src: 2123, dest: 30016) to the S5-GTP-EP.

Step	Description
9	The S5-GTP-EP pod sends MBCmd-FailureIndication to SGW-SVC pod to end the transaction. Post internal transaction, the GRPC Update Bearer Request is sent to the SGW-SVC pod.
10	The S5-GTP-EP starts the ET2.
11	The S5-GTP-EP sends the GRPC Update Bearer Request to SGW-SVC pod.
12	The SGW-SVC pod processes the Update Bearer Request and consumes the saved MBC_Info to send Update Bearer Request to the S11-GTP-EP.
13	The S11-GTP-EP sends the Update Bearer Request to the S11-UDP-Proxy.
14	The S11-UDP-Proxy forwards the Update Bearer Request to the MME.
15	The MME sends Update Bearer Response to the SGW-SVC pod.
16	The S11-UDP-Proxy processes and sends the Update Bearer Response to the SGW-SVC pod.
17	The SGW-SVC pods forward the Update Bearer Response to the S5-GTP-EP pod.
18	The S5-GTP-EP pod sends the Update Bearer Response to the S5-UDP-Proxy.
19	The S5-UDP-Proxy sends the Update Bearer Response to the PGW.

DBC Failure Handling Call Flow

This section describes the DBC Failure Handling call flow.

Figure 91: DBC Failure Handling Call Flow

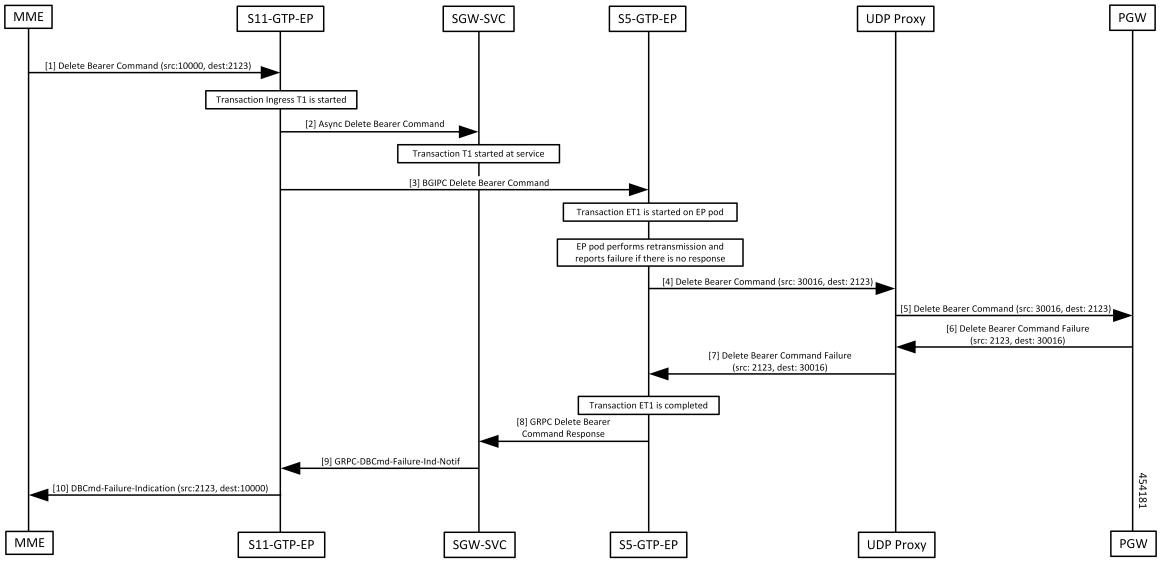


Table 169: DBC Failure Handling Call Flow Description

Step	Description
1	The MME sends the Delete Bearer Command to the S11-GTP-EP pod.
2	The S11-GTP-EP pod sends the ASYNC Delete Bearer Command to the SGW-SVC pod.
3	The SGW-SVC pod forwards Delete Bearer Command to the PGW. Save DBC_info in PDN for response. The EP pod performs retransmission. If there is no response, the pod sends the Delete Bearer Command Failure Indication (DBCFI) to the SGW-SVC pod.
4	The S5-GTP-EP sends the Delete Bearer Command to the UDP Proxy.
5	The UDP Proxy forwards the Delete Bearer Command to the PGW.
6	The PGW sends the Delete Bearer Command Failure to the UDP Proxy. DBCFI is received on S5 GTPC-EP pod and is forwarded to the SGW-SVC pod.
7	The UDP Proxy forwards the Delete Bearer Command Failure to the S5-GTP-EP.
8	The S5-GTP-EP sends the GRPC Delete Bearer Command Response to the SGW-SVC.
9	The SGW-SVC pod sends the GRPC-DBCmd-Failure-Ind-Notif to the S11-GTP-EP.
10	The S11-GTP-EP pod processes the DBCFI and forwards the response to MME with the saved DBC_Info.

DBC Success Handling Call Flow

This section describes the DBC Success Handling call flow.

Figure 92: DBC Success Handling Call Flow

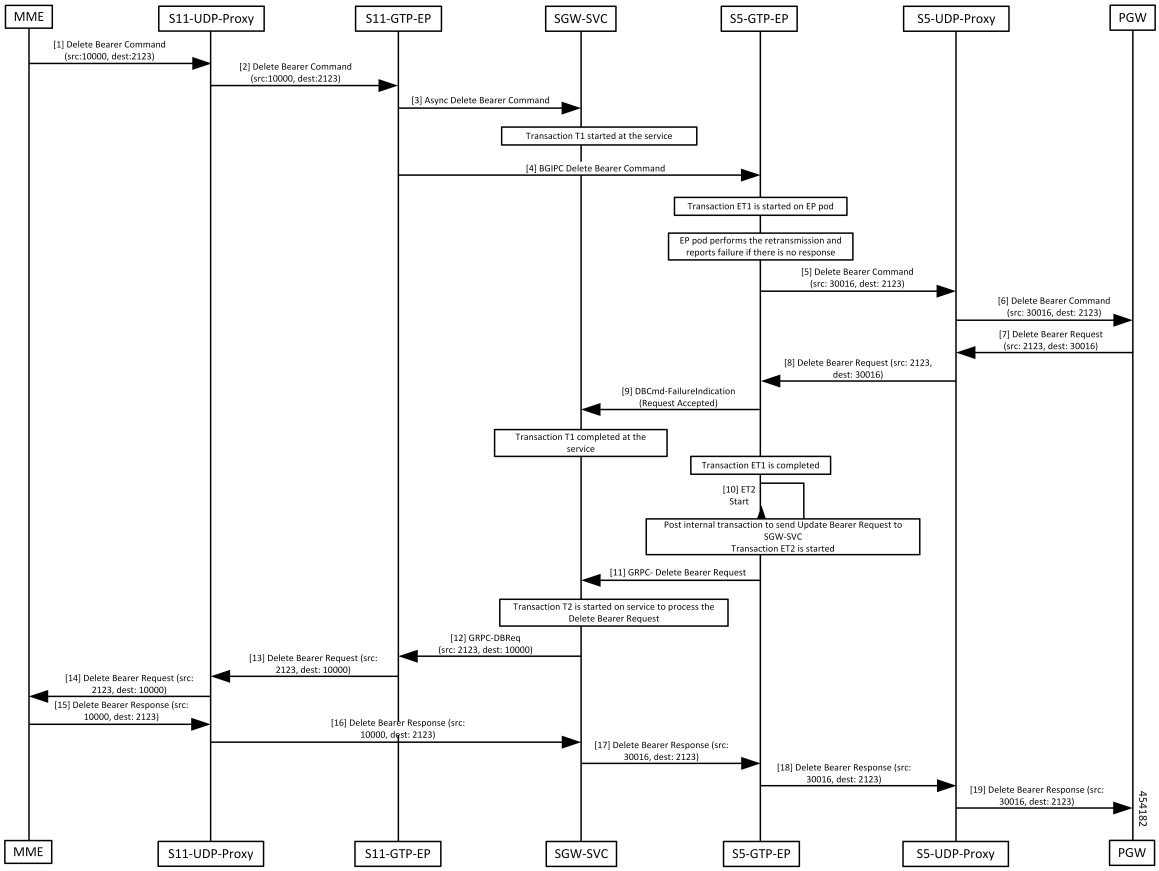
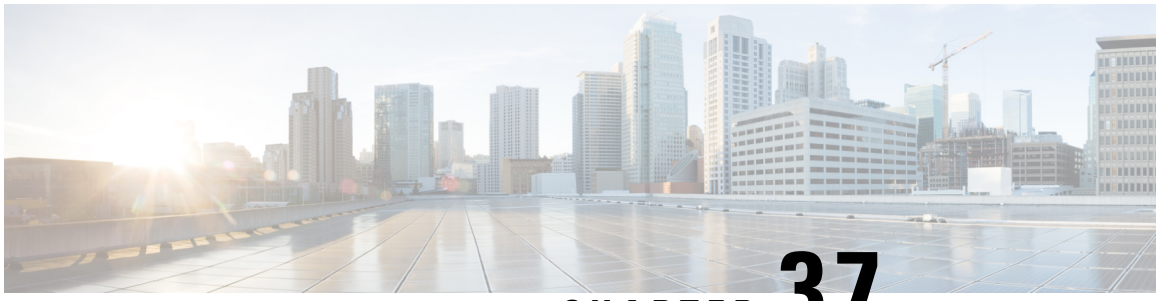


Table 170: DBC Success Handling Call Flow Description

Step	Description
1	The MME sends a Delete Bearer Command to the S11-UDP-Proxy.
2	The S11-UDP-Proxy forwards the Delete Bearer Command to the S11-GTP-EP.
3	The S11-UDP-Proxy sends the ASYNC Delete Bearer Command to the SGW-SVC pod.
4	The SGW-SVC pod sends the BGIPC Delete Bearer Command to the S5-GTP-EP. Save DBC_info in the PDN for response.
5	The EP pod performs the retransmission and reports a failure if there is no response. The pod sends the Delete Bearer Command Failure Indication (DBCFI) to the SGW-SVC pod. The S5-GTP-EP sends the Delete Bearer Command to the S5-UDP-Proxy.
6	The S5-UDP-Proxy send the Delete Bearer Command to the PGW.
7	The PGW sends the Delete Bearer Request to the S5-UDP-Proxy pod.
8	The S5-UDP-Proxy pod forwards the Delete Bearer Request to the S5-GTP-EP.

Step	Description
9	The S5-GTP-EP pod sends the DBCFI (with Request as ACCEPTED) to the SGW-SVC pod to end the transaction. The SGW-SVC pod ends the transaction and consumes this DBCFI. The post internal transaction sends the GRPCE_DBReq to SGW-SVC pod.
10	After the ET1 transaction is completed, the S5-GTP-EP starts.
11	The S5-GTP-EP pod sends the GRPC-DBRequest to the SGW-SVC.
12	The SGW-SVC pod processes the Delete Bearer Request and used saved DBC_Info to send the Updated Bearer Request to the MME.
13	The S11-GTP-EP pod sends the Delete Bearer Request to the S11-UDP-Proxy pod.
14	The S11-UDP-Proxy forwards the Delete Bearer Request to the MME.
15	The MME sends the Delete Bearer Response to the S11-UDP-Proxy pod.
16	The S11-UDP-Proxy processes the Delete Bearer Response to the SGW-SVC.
17	The SGW-SVC forwards the Delete Bearer Response to the S5-GTP-EP.
18	The S5-GTP-EP pod sends the Delete Bearer Response to the S5-UDP-Proxy.
19	The S5-UDP-Proxy sends the Delete Bearer Response to the PGW.



CHAPTER 37

Modify Bearer Request Support

- Feature Summary and Revision History, on page 391
- Feature Description, on page 391
- How it Works, on page 392

Feature Summary and Revision History

Summary Data

Table 171: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Revision Details	Release
First introduced.	2020.03.0

Feature Description

cnSGW-C supports the MBR service request from MME to change the UE state from IDLE to ACTIVE. cnSGW-C supports the following service requests:

- UE-triggered service request without PGW interaction
- UE-triggered service request with PGW interaction

How it Works

This section describes how this feature works.

The cnSGW-C performs the following actions while processing the UE-triggered service request:

- Sends the Sx Modification Request message to the UPF to:
  - Mark downlink Forwarding Action Rule (FAR) as forward.
  - Update the S1 eNodeB-F TEID information to UPF sends the downlink packets to eNodeB.
- After receiving the Sx Modify Response message from the UPF, cnSGW-C:
  - Sends the Modify Bearer Response message to MME.
  - Checks User Location Information (ULI) or UE time zone. For any change in the time zone, it sends Modify Bearer Request to PGW to update the TAI. The UE-triggered service request with PGW interaction request only considers ULI or UE time zone check.

Call Flows

This section describes the key call flows for this feature.

UE-Triggered Service Request without PGW Interaction Call Flow

This section describes the UE-Triggered Service Request without PGW Interaction call flow.

Figure 93: UE Triggered Service Request without PGW Interaction Call Flow

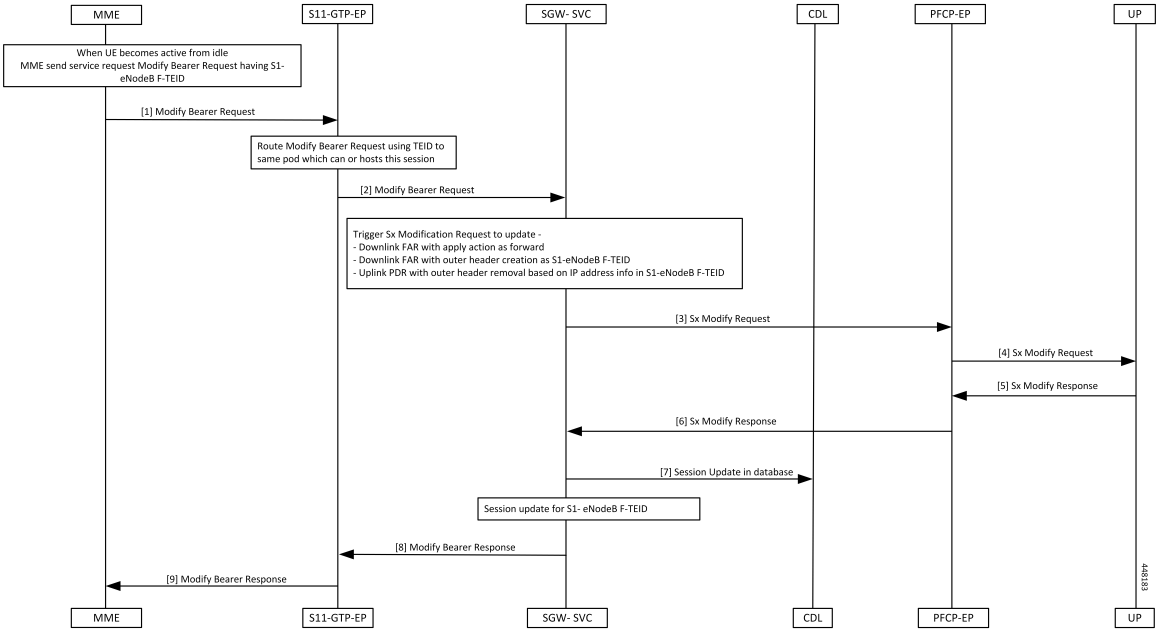


Table 172: UE Triggered Service Request without PGW Interaction Call Flow Description

Step	Description
1	The MME sends the Modify Bearer Request message with s1-eNodeB F-TEID to cnSGW-C when the UE changes from IDLE to ACTIVE state.
2	The S11-GTP-EP decodes the UDP message and converts it into gRPC message. This gRPC message is sent to the SGW-SVC pod (which can handle this UE session) using TEID.
3	The SGW-SVC pod finds the subscriber context using the local ingress TEID. The SGW-SVC pod sends the Modify Bearer Request content and sends Sx Modify Request to PFCP-EP.
4	The PFCP-EP sends Sx Modify Request message to UPF through the UDP proxy.
5	The UPF process the Sx Modify Request message and sends Sx Modify Response message to PFCP-EP.
6	The PFCP-EP sends the Sx Modify Response message to SGW-SVC pod.
7	The SGW-SVC pod chnages PDN into CONNECTED state and sends session update to CDL. The CDL module updates the information in the database.
8	The SGW-SVC pod sends the Modify Bearer Response message to the S11-GTP-EP.
9	The S11-GTP-EP sends the Modify Bearer Response message to MME. The MME processes the Modify Bearer Response message.

UE-Triggered Service Request with PGW Interaction Call Flow

This section describes the UE-Triggered Service Request with PGW Interaction call flow.

Figure 94: UE-Triggered Service Request with PGW Interaction Call Flow

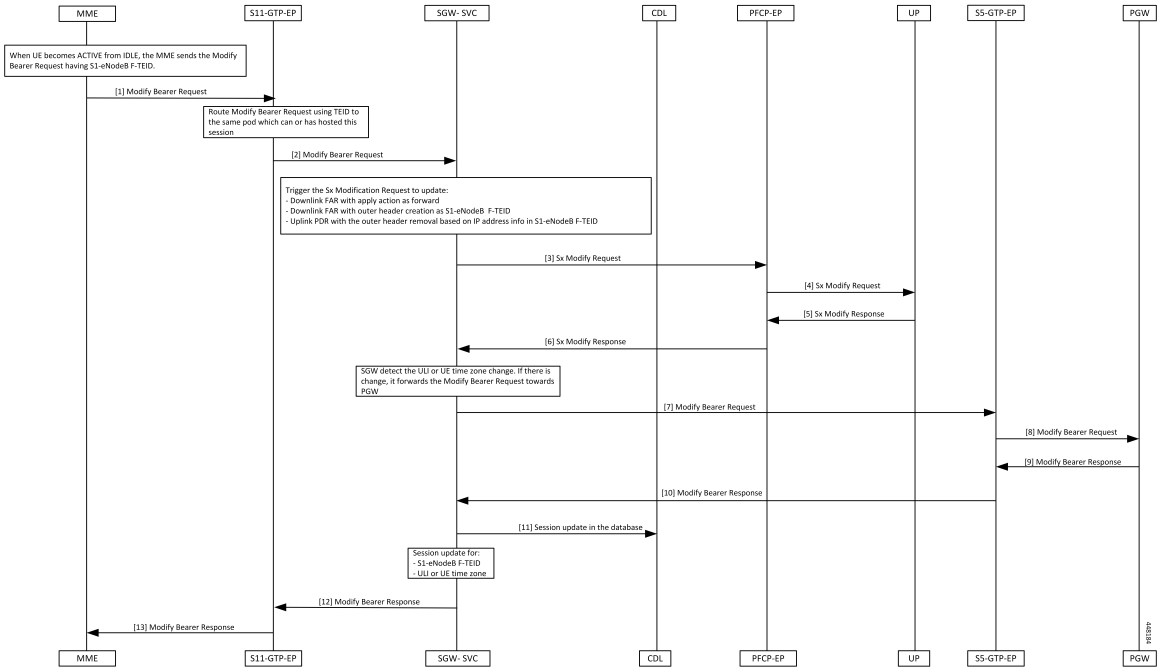
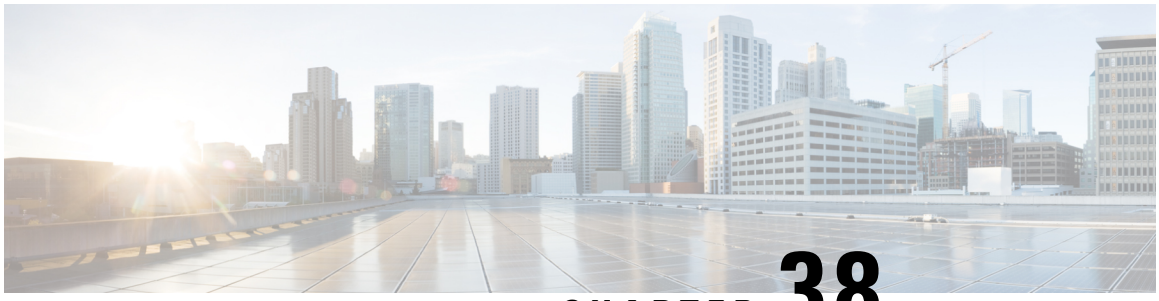


Table 173: UE-Triggered Service Request with PGW Interaction Call Flow description

Step	Description
1	The MME sends the Modify Bearer Request with s1-eNodeB F-TEID to cnSGW-C when the UE changes from the IDLE to ACTIVE state.
2	The S11-GTP-EP decodes the UDP message and converts it into the gRPC message. This gRPC message is sent to the SGW-Service pod, which handles the UE session using TEID.
3	The SGW-Service pod finds the subscriber context using the local ingress TEID. It validates the Modify Bearer Request content and sends the Sx Modify Request to PFCP-EP.
4	The PFCP-EP sends the Sx Modify Request to the UPF through the UDP proxy.
5	The UPF1 processes the Sx Modify Request and sends the Sx Modify Response message.
6	The PFCP-EP sends the Sx Modify Response message to the SGW-Service pod.
7	The SGW-Service pod detects ULI or UE time zone change and sends the Modify Bearer Request message to S5-GTP-EP.
8	The S5-GTP-EP sends the Modify Bearer Request message to the PGW.
9	The PGW processes the Modify Bearer Request message and sends the Modify Bearer Response message.
10	The S5-GTP-EP sends the Modify Bearer Response message to the SGW-Service pod.

Step	Description
11	The SGW-Service pod moves PDN into the CONNECTED state and sends the update to CDL. The CDL module updates the information in the database.
12	The SGW-Service pod sends the Modify Bearer Response message to the S11-GTP-EP.
13	The S11-GTP-EP sends the Modify Bearer Response message to the MME. The MME processes the Modify Bearer Response message.



CHAPTER 38

Monitor Subscriber and Protocol Support

- Feature Summary and Revision History, on page 397
- Feature Description, on page 397
- Feature Configuration, on page 398

Feature Summary and Revision History

Summary Data

Table 174: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled – Configuration required to disable
Related Documentation	Not Applicable

Revision History

Table 175: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

The cnSGW-C service supports the subscriber map and the operator policy configurations for the SGW service parameters.

Feature Configuration

Configuring this feature involves the following steps:

- Configure the monitor subscriber utility to trace messages related to a specified subscriber. For more information, refer to Configuring the Monitor Subscriber, on page 398.
- An example of the monitor subscriber utility. For more informatio, refer to Configuration Example, on page 399.
- Configure the monitor protocol utility to capture packets on a specified interface. For more informatio, refer toConfiguring the Monitor Protocol, on page 421.
- An example of the monitor protocol utility. For more informatio, refer to Configuration Example, on page 422.
- Configure the Request Response messages in the transaction logs. For more information, refer to Configuring the Transaction Messages, on page 433.
- An example of the transaction logs. For more information, refer to Configuration Example, on page 433.
- Access the monitor subscriber and protocol logs. For more information, refer to Accessing the Logs, on page 437

Configuring the Monitor Subscriber

To configure this feature use the following configuration:

```
exec
   monitor subscriber
      capture-duration capture_duration
      dump filename filename_value
      gr-instance gr_instance
      imei imei_value
      imsi imsi_value
      supi supi_value
      list subscriber_list
      internal-messages [ Yes | No ]
      transaction-logs [ Yes | No ]
      nf-service nf_service
      end
```

Note In 2021.02 and later releases, the namespace keyword is deprecated and replaced with nf-service.

NOTES:

- **capture-durationi** *capture_duration*—Specify the duration in seconds during which the monitor subscriber feature is enabled. The default value is 300 seconds.
- **supi** *supi_value*—Specify the subscriber identifier. For example, imsi-123456789 and imsi-123*

- **imsi** *imsi_value*—Specify the IMSI value. For example, 123456789 and *
- **imei** *imei_value*—Specify the IMEI value. For example, 123456789012345 and *
- **internal-messages [Yes | No]**—Configures internal messaging. When set to yes, the internal messaging is enabled. By default, the configuration is disabled.
- **transaction-logs [Yes | No]** —Configures transaction logging. By default, the configuration is disabled.

Note At any point, either the internal messages or the transaction logs are displayed.

- **nf-service** *nf_service*—Specify the NF service. The accepted services are sgw and smf. The default value is none.
- **gr-instance** *gr_instance*—Specify the GR instance that the cnSGW-C monitors the subscriber for.

Configuration Example

The following is an example configuration.

```
monitor subscriber imsi 123456789 capture-duration 100 internal-messages yes
monitor subscriber imsi 123456789 capture-duration 100 transaction-logs yes
```

Sample Output

The following is a sample output.

```
monitor subscriber imsi * namespace sgw
supi: imsi-*
captureDuraiton: 300
enableInternalMsg: false
enableTxnLog: false
namespace(deprecated. Use nf-service instead.): sgw
nf-service: none
gr-instance: 0
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   277  100    89  100   188   5235  11058 --:--:-- --:--:-- --:--:-- 16294
Command: --header Content-type:application/json --request POST --data
{"commandname":"mon_sub","parameters":{"supi":"imsi-*","duration":300,
"enableTxnLog":false,
"enableInternalMsg":false,"action":"start","namespace":"sgw","nf-service":"none",
"grInstance":0}}
http://oam-pod:8879/commands
Result start mon_sub, fileName ->
logs/monsublogs/sgw.imsi-*_TS_2021-08-15T12:36:17.569800845.txt
Starting to tail the monsub messages from file:
logs/monsublogs/sgw.imsi-*_TS_2021-08-15T12:36:17.569800845.txt
Defaulting container name to oam-pod.
Use 'kubectl describe pod/oam-pod-0 -n cn' to see all of the containers in this pod.
      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.288997
      Message: Sx Session Establishment Request
      Description: Sx Session Establishment Request Message from SGWC to SGWU
      Source: 209.165.201.19 (SGW.protocol.DC.Local.1)
      Destination: 209.165.201.20 (SGW.udp-proxy.DC.Local.0)
      PAYLOAD:
        Sx Session Establishment Request:
```

```
Sx Session Establishment Request:
    FSeid:
        Seid: 1297038098512740679
        IPv4Address: 209.165.201.19
    CreatePdr:
        CreatePdr[0]:
            PdrId: 1
            Precedence: 0
            Pdi:
                SrcIf: CORE
                UeIp:
                    Src: false
                    Dst: false
                    IPv4Addr: 209.165.201.30
                TEndpointId: 1
                Valid: true
            OuterHdrRem: 0
            FarId:
                FarId[0]: 1
            Qfi: 0
            OuterHdrRemValid: false
        CreatePdr[1]:
            PdrId: 2
            Precedence: 0
            Pdi:
                SrcIf: ACCESS
                UeIp:
                    Src: false
                    Dst: false
                    IPv4Addr: 209.165.201.30
                TEndpointId: 2
                Valid: true
            OuterHdrRem: 0
            FarId:
                FarId[0]: 2
            Qfi: 0
            OuterHdrRemValid: false
    CreateFar:
        CreateFar[0]:
            FarId: 1
            ApplyAction:
                Drop: true
                Frwd: false
                Buff: false
                Nocp: false
                Dupl: false
                Valid: true
            FwdParams:
                DestIf: ACCESS
                RedirectInfo:
                    AddrType: 0
                    Valid: false
                OuterHdr:
                    OuterHdrDesc: 0
                    Teid: 0
                    IPv4Address: 209.165.201.30
                    Port: 0
                    Valid: false
                TEndptId: 2
                OuterPktTos: 255
                InnerPktTos: 255
                TosOpt:
                    CopyInner: false
                    CopyOuter: false
```

```
            SendTos: 0
            PfcpSmFlags:
                Drobu: false
                Qaurr: false
                Sndem: false
                Valid: false
            Valid: true
            NextHopId: 0
        DuplParams:
            DestIf: ACCESS
            OuterHdr:
                OuterHdrDesc: 0
                Teid: 0
                IPv4Address: 209.165.201.30
                Port: 0
                Valid: false
            InterceptInfo:
                InterceptId: 0
                ChargingId: 0
                SmfLiNodeId:
                    IpDesc: 0
                    IPv4Address: 209.165.201.30
                    Valid: false
                PduSessionId: 0
                Valid: false
            Valid: false
        BarId: 0
    CreateFar[1]:
        FarId: 2
        ApplyAction:
            Drop: true
            Frwd: false
            Buff: false
            Nocp: false
            Dupl: false
            Valid: true
        FwdParams:
            DestIf: CORE
            RedirectInfo:
                AddrType: 0
                Valid: false
            OuterHdr:
                OuterHdrDesc: 0
                Teid: 0
                IPv4Address: 209.165.201.30
                Port: 0
                Valid: false
            TEndptId: 1
            OuterPktTos: 255
            InnerPktTos: 255
            TosOpt:
                CopyInner: false
                CopyOuter: false
            SendTos: 0
            PfcpSmFlags:
                Drobu: false
                Qaurr: false
                Sndem: false
                Valid: false
            Valid: true
            NextHopId: 0
        DuplParams:
            DestIf: ACCESS
            OuterHdr:
```

```
                OuterHdrDesc: 0
                Teid: 0
                IPv4Address: 209.165.201.30
                Port: 0
                Valid: false
            InterceptInfo:
                InterceptId: 0
                ChargingId: 0
                SmfLiNodeId:
                    IpDesc: 0
                    IPv4Address: 209.165.201.30
                    Valid: false
                PduSessionId: 0
                Valid: false
            Valid: false
        BarId: 0
CreateTEndpt:
    CreateTEndpt[0]:
        EndpointId: 1
        FTeid:
            Teid: 0
            IPv4Address: 209.165.201.30
            ChooseId: 0
        BearerLvlInfo:
            Valid: 1
            Qci: 6
    CreateTEndpt[1]:
        EndpointId: 2
        FTeid:
            Teid: 0
            IPv4Address: 209.165.201.30
            ChooseId: 0
        BearerLvlInfo:
            Valid: 1
            Qci: 6
PdnType: 0
UplaneInacTimer: 0
MetaData: From:209.165.201.19:10665->To:209.165.201.20:8805
Supi:
Seid: 1297038098512740679
Seqno: 4252
Version: 0
MsgPriority: false
MsgPriorityVal: 0
Cmnid: 0
Rseid: 0
IntfType: 0
HdrLen: 0
MsgLen: 0
UserIDInfo:
    Imsi: 123456789012348
    Imei: 123456786666660
    Msisdn: 223310101010101
    Valid: true
XHeaderInfo:
    RatType:
    Valid: false
CfPolicyId:
    PolicyId: 0
    Valid: false
ChargingDisabled:
    Valid: false
    Value: false
ChargingParams:
```

```
                    Valid: 0
                    GyOfflineChargingEnabled: 0
                NextHopIPv4: 0

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.371114
      Message: Sx Session Establishment Response
      Description: Sx Session Establishment Response Message from SGWU to SGWC
      Source: 209.165.201.20 (SGW.udp-proxy.DC.Local.0)
      Destination: 209.165.201.19 (SGW.protocol.DC.Local.1)
      PAYLOAD:
        Sx Session Establishment Response:
            Sx Session Establishment Response:
                Cause: 1
                OffendingIe: 0
                FSeid:
                    Seid: 10002
                    IPv4Address: 209.165.201.20
                CreatedTEndpt:
                    CreatedTEndpt[0]:
                        EndpointId: 1
                        FTeid:
                            Teid: 25270
                            IPv4Address: 209.165.201.1
                            ChooseId: 0
                    CreatedTEndpt[1]:
                        EndpointId: 2
                        FTeid:
                            Teid: 25271
                            IPv4Address: 209.165.200.226
                            ChooseId: 0
                MetaData: From:209.165.201.20:8805->To:209.165.201.19:10665
                Supi:
                Seid: 1297038098512740679
                Seqno: 4252
                Version: 0
                MsgPriority: false
                MsgPriorityVal: 0
                Cmnid: 0
                Rseid: 0
                IntfType: 0
                HdrLen: 0
                MsgLen: 0
                LoadControlInfo:
                    SeqNum: 0
                    Metric: 0
                    Valid: false
                OverloadControlInfo:
                    SeqNum: 0
                    Metric: 0
                    Ociflag: 0
                    Valid: false

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.377609
      Message: GtpEpDecodeRPCResponse
      Description:  2071
      Source:
      Destination:
      PAYLOAD:
```

```
-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.379043
      Message: S5 S8 Create Session Request
      Description: S5 S8 Create Session Request Message
      Source: 209.165.201.19
      Destination: 209.165.201.18
      PAYLOAD:
        S5 S8 Create Session Request:
            S5 S8 Create Session Request:
                Version: 2
                Pflag: false
                TEIDflag: true
                MsgPriority: false
                MsgLength: 221
                TIED: 0
                Seq: 66683
                MsgTypeId: 32
                MsgPriorityValue: 0
                Peer_IPv4_Flag: false
                Peer_IPv6_Flag: false
                MetaData: From:209.165.201.19:15001->To:209.165.201.18:2123
                Seid: 0
                Rseid: 0
                Cmnid: 0
                MsgType:
                    Create_Session_Request:
                        IMSI: 123456789012348
                        Recovery:
                            Value: 0
                        APN: intershat
                        AMBR:  UL: 232323 kbps, DL: 232323 kbps
                        MEI: 123456786666660
                        MSISDN: 223310101010101
                        Indication:
                            DAF: false
                            DTF: false
                            HI: false
                            DFI: false
                            OI: false
                            ISRSI: false
                            ISRAI: false
                            SGWCI: false
                            SQCI: false
                            UIMSI: false
                            CFSI: false
                            CRSI: false
                            P: false
                            PT: false
                            SI: false
                            MSV: false
                            RetLoc: false
                            PBIC: false
                            SRNI: false
                            S6AF: false
                            S4AF: false
                            MBMDT: false
                            ISRAU: false
                            CCRSI: false
                            CPRAI: false
                            ARRL: false
                            PPOF: false
```

```
    PPON_PPEI: false
    PPSI: false
    CSFBI: false
    CLII: false
    CPSR: false
    NSI: false
    UASI: false
    DTCI: false
    BDWI: false
    PSCI: false
    PCRI: false
    AOSI: false
    AOPI: false
    ROAAI: false
    EPCOSI: false
    CPOPCI: false
    PMTSMI: false
    S11TF: false
    PNSI: false
    UNACCSI: false
    WPMSI: false
    5GSIWK: false
    EEVRSI: false
    LTEMUI: false
    LTEMPI: false
    ENBCRSI: false
    TSPCMI: false
    PGBK: false
    PCPSI: false
    PCP: false
    PCPU: false
    N26_5GS: false
    RI_5GCN: false
    RS_5GCN: false
PAA:
    PDN_Type: 1
    IPv4: 209.165.201.30
    IPv6_Prefix: 0
RAT_Type:
    Value: 6
Serving_Network:
    MCC: 123
    MNC: 456
ULI:
    UliTai:  Mcc: 123, Mnc: 456, TAC: 2346
    UliEcgi:
        Mcc: 123
        Mnc: 456
        Eci: 1234567
FQ_TEID:
    SgwCntrl:
        IFace: 6
        TEID: 1375732039
        IPv4: 209.165.201.19
Bearer_Context_List:
    NumBearerCtxt: 1
    PbBearerCxt:
        PbBearerCxt[0]:
            BearerCtxType: 0
            EBI: 5
            Fqteid:
                SgwData:
                    IFace: 0
                    TEID: 25270
```

```
                                        IPv4: 209.165.201.1
                                BearerQos:
                                    PCI: true
                                    PL: 12
                                    PVI: true
                                    QCI: 6
                                    UL_MBR: 0 kbps
                                    DL_MBR: 0 kbps
                                    UL_GBR: 0 kbps
                                    DL_GBR: 0 kbps
                                    Arp: 113
                                    QciType: 0
                            Charging_Characteristics:
                                Value:
                                    Value[0]: 210
                                    Value[1]: 4
                                    Value[2]: 0
                                    Value[3]: 0
                            PDN_Type:
                                Value: 1
                            UE_Time_Zone:
                                Time_Zone: 16
                                Daylight_Saving_Time: 1
                            APN_Restriction:
                                Value: 0
                            Selection_Mode:
                                Value: 0
                            EPCO:
                                Len: 5
                                Value:
                                    Value[0]: 128
                                    Value[1]: 0
                                    Value[2]: 26
                                    Value[3]: 1
                                    Value[4]: 5

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/06/08 04:37:51.487884
      Message: GtpEpDecodeRPCResponse
      Description:  2071
      Source:
      Destination:
      PAYLOAD:

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/06/08 04:37:51.487884
      Message: GtpEpDecodeRPCIPCResponse
      Description:  2071
      Source:
      Destination:
      PAYLOAD:

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.475859
      Message: GtpEpDecodeRPCResponse
      Description:  2071
      Source:
      Destination:
```

```
      PAYLOAD:

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.476252
      Message: S5 S8 Create Session Response
      Description: S5 S8 Create Session Response Message
      Source: 209.165.201.18
      Destination: 209.165.201.19
      PAYLOAD:
        S5 S8 Create Session Response:
            S5 S8 Create Session Response:
                Version: 2
                Pflag: false
                TEIDflag: true
                MsgPriority: false
                MsgLength: 107
                TIED: 1375732039
                Seq: 66683
                MsgTypeId: 33
                MsgPriorityValue: 0
                Peer_IPv4_Flag: false
                Peer_IPv6_Flag: false
                MetaData: From:209.165.201.18:0->To:209.165.201.19:0
                Seid: 0
                Rseid: 0
                Cmnid: 0
                MsgType:
                    Create_Session_Response:
                        Cause:
                            Cause_Value: 16
                            PCE: false
                            BCE: false
                            OrigInd: false
                        Recovery:
                            Value: 100
                        AMBR:  UL: 10 kbps, DL: 20 kbps
                        PAA:
                            PDN_Type: 1
                            IPv4: 209.165.201.26
                            IPv6_Prefix: 0
                        FQ_TEID:
                            PgwCntrl:
                                IFace: 7
                                TEID: 13210
                                IPv4: 209.165.201.18
                        Bearer_Context_List:
                            NumBearerCtxt: 1
                            PbBearerCxt:
                                PbBearerCxt[0]:
                                    BearerCtxType: 0
                                    EBI: 5
                                    Cause:
                                        Cause_Value: 16
                                        PCE: false
                                        BCE: false
                                        OrigInd: false
                                    Fqteid:
                                        PgwData:
                                            IFace: 5
                                            TEID: 13211
                                            IPv4: 209.165.201.18
                                    ChrgId:
```

```
                                        Value: 303174163
                            APN_Restriction:
                                Value: 1

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.483195
      Message: Sx Session Modification Request
      Description: Sx Session Modification Request Message from SGWC to SGWU
      Source: 209.165.201.19 (SGW.protocol.DC.Local.2)
      Destination: 209.165.201.20 (SGW.udp-proxy.DC.Local.0)
      PAYLOAD:
        Sx Session Modification Request:
            Sx Session Modification Request:
                UpdatePdr:
                    UpdatePdr[0]:
                        PdrId: 1
                        OuterHdrRem: 0
                        Precedence: 0
                        Pdi:
                            SrcIf: ACCESS
                            UeIp:
                                Src: false
                                Dst: false
                                IPv4Addr: 209.165.201.30
                            TEndpointId: 0
                            Valid: false
                        Qfi: 0
                UpdateFar:
                    UpdateFar[0]:
                        FarId: 2
                        ApplyAction:
                            Drop: false
                            Frwd: true
                            Buff: false
                            Nocp: false
                            Dupl: false
                            Valid: true
                        UpdateFwdParams:
                            DestIf: ACCESS
                            RedirectInfo:
                                AddrType: 0
                                Valid: false
                            OuterHdr:
                                OuterHdrDesc: 256
                                Teid: 13211
                                IPv4Address: 209.165.201.18
                                Port: 0
                                Valid: true
                            TEndptId: 0
                            OuterPktTos: 0
                            InnerPktTos: 0
                            TosOpt:
                                CopyInner: false
                                CopyOuter: false
                            SendTos: 0
                            PfcpSmFlags:
                                Drobu: false
                                Qaurr: false
                                Sndem: false
                                Valid: false
                            Valid: true
                            NextHopId: 0
```

```
                    UpdateDuplParams:
                        DestIf: ACCESS
                        OuterHdr:
                            OuterHdrDesc: 0
                            Teid: 0
                            IPv4Address: 209.165.201.30
                            Port: 0
                            Valid: false
                        InterceptInfo:
                            InterceptId: 0
                            ChargingId: 0
                            SmfLiNodeId:
                                IpDesc: 0
                                IPv4Address: 209.165.201.30
                                Valid: false
                            PduSessionId: 0
                            Valid: false
                        Valid: false
                    BarId: 0
                UplaneInacTimer: 0
                MetaData: From:209.165.201.19:10002->To:209.165.201.20:8805
                Supi:
                Seid: 1297038098512740679
                Seqno: 4248
                Version: 0
                MsgPriority: false
                MsgPriorityVal: 0
                Cmnid: 0
                Rseid: 10002
                IntfType: 0
                HdrLen: 0
                MsgLen: 0
                PfcpSmFlags:
                    Drobu: false
                    Qaurr: false
                    Sndem: false
                    Valid: false
                UserIDInfo:
                    Valid: false
                XHeaderInfo:
                    RatType:
                    Valid: false
                CfPolicyId:
                    PolicyId: 0
                    Valid: false
                GyStatus:
                    Valid: false
                    Value: false
                ChargingDisabled:
                    Valid: false
                    Value: false
                QueryInterface:
                    Valid: false
                    OfflineUrr: false
                    OnlineUrr: false
                    RadiusUrr: false
                    BearerUrr: false
                    SessUrr: false

------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.549171
      Message: Sx Session Modification Response
```

```
        Description: Sx Session Modification Response Message from SGWU to SGWC
        Source: 209.165.201.20 (SGW.udp-proxy.DC.Local.0)
        Destination: 209.165.201.19 (SGW.protocol.DC.Local.2)
        PAYLOAD:
          Sx Session Modification Response:
              Sx Session Modification Response:
                  Cause: 1
                  OffendingIe: 0
                  LoadControlInfo:
                      SeqNum: 0
                      Metric: 0
                      Valid: false
                  OverloadControlInfo:
                      SeqNum: 0
                      Metric: 0
                      Ociflag: 0
                      Valid: false
                  MetaData: From:209.165.201.20:8805->To:209.165.201.19:10002
                  Supi:
                  Seid: 1297038098512740679
                  Seqno: 4248
                  Version: 1
                  MsgPriority: false
                  MsgPriorityVal: 0
                  Cmnid: 0
                  Rseid: 0
                  IntfType: 0
                  HdrLen: 17
                  MsgLen: 0

------------------------------------------------

        Subscriber Id: imsi-123456789012348
        Timestamp: 2021/08/15 12:39:25.262224
        Message: S11 Create Session Request
        Description: S11 Create Session Request Message
        Source: 209.165.201.20
        Destination: 209.165.201.19
        PAYLOAD:
          S11 Create Session Request:
              S11 Create Session Request:
                  Version: 2
                  Pflag: false
                  TEIDflag: true
                  MsgPriority: false
                  MsgLength: 223
                  TIED: 0
                  Seq: 5842
                  MsgTypeId: 32
                  MsgPriorityValue: 0
                  Peer_IPv4_Flag: false
                  Peer_IPv6_Flag: false
                  MetaData: From:209.165.201.20:2123->To:209.165.201.19:2123
                  Seid: 0
                  Rseid: 0
                  Cmnid: 0
                  MsgType:
                      Create_Session_Request:
                          IMSI: 123456789012348
                          Recovery:
                              Value: 100
                          APN: intershat
                          AMBR:  UL: 232323 kbps, DL: 232323 kbps
                          MEI: 123456786666660
```

```
MSISDN: 223310101010101
Indication:
    DAF: false
    DTF: false
    HI: false
    DFI: false
    OI: false
    ISRSI: false
    ISRAI: false
    SGWCI: false
    SQCI: false
    UIMSI: false
    CFSI: false
    CRSI: false
    P: false
    PT: false
    SI: false
    MSV: false
    RetLoc: false
    PBIC: false
    SRNI: false
    S6AF: false
    S4AF: false
    MBMDT: false
    ISRAU: false
    CCRSI: false
    CPRAI: false
    ARRL: false
    PPOF: false
    PPON_PPEI: false
    PPSI: false
    CSFBI: false
    CLII: false
    CPSR: false
    NSI: false
    UASI: false
    DTCI: false
    BDWI: false
    PSCI: false
    PCRI: false
    AOSI: false
    AOPI: false
    ROAAI: false
    EPCOSI: false
    CPOPCI: false
    PMTSMI: false
    S11TF: false
    PNSI: false
    UNACCSI: false
    WPMSI: false
    5GSIWK: false
    EEVRSI: false
    LTEMUI: false
    LTEMPI: false
    ENBCRSI: false
    TSPCMI: false
    PGBK: false
    PCPSI: false
    PCP: false
    PCPU: false
    N26_5GS: false
    RI_5GCN: false
    RS_5GCN: false
PAA:
```

```
    PDN_Type: 1
    IPv4: 209.165.201.30
    IPv6_Prefix: 0
RAT_Type:
    Value: 6
Serving_Network:
    MCC: 123
    MNC: 456
ULI:
    UliTai:  Mcc: 123, Mnc: 456, TAC: 2346
    UliEcgi:
        Mcc: 123
        Mnc: 456
        Eci: 1234567
FQ_TEID:
    MmeCntrl:
        IFace: 10
        TEID: 25269
        IPv4: 209.165.201.20
    PgwCntrl:
        IFace: 7
        TEID: 0
        IPv4: 209.165.201.18
Bearer_Context_List:
    NumBearerCtxt: 1
    PbBearerCxt:
        PbBearerCxt[0]:
            BearerCtxType: 0
            EBI: 5
            Fqteid:
            BearerQos:
                PCI: true
                PL: 12
                PVI: true
                QCI: 6
                UL_MBR: 0 kbps
                DL_MBR: 0 kbps
                UL_GBR: 0 kbps
                DL_GBR: 0 kbps
                Arp: 113
                QciType: 0
Charging_Characteristics:
    Value:
        Value[0]: 210
        Value[1]: 4
        Value[2]: 0
        Value[3]: 0
PDN_Type:
    Value: 1
UE_Time_Zone:
    Time_Zone: 16
    Daylight_Saving_Time: 1
APN_Restriction:
    Value: 0
Selection_Mode:
    Value: 0
EPCO:
    Len: 5
    Value:
        Value[0]: 128
        Value[1]: 0
        Value[2]: 26
        Value[3]: 1
        Value[4]: 5
```

```
-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.568954
      Message: GtpEpDecodeRPCResponse
      Description:  258
      Source:
      Destination:
      PAYLOAD:

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.569939
      Message: S11 Create Session Response
      Description: S11 Create Session Response Message
      Source: 209.165.201.19
      Destination: 209.165.201.20
      PAYLOAD:
        S11 Create Session Response:
            S11 Create Session Response:
                Version: 2
                Pflag: false
                TEIDflag: true
                MsgPriority: false
                MsgLength: 120
                TIED: 25269
                Seq: 5842
                MsgTypeId: 33
                MsgPriorityValue: 0
                Peer_IPv4_Flag: false
                Peer_IPv6_Flag: false
                MetaData: From:209.165.201.19:2123->To:209.165.201.20:2123
                Seid: 0
                Rseid: 0
                Cmnid: 0
                MsgType:
                    Create_Session_Response:
                        Cause:
                            Cause_Value: 16
                            PCE: false
                            BCE: false
                            OrigInd: false
                        Recovery:
                            Value: 0
                        AMBR:  UL: 10 kbps, DL: 20 kbps
                        PAA:
                            PDN_Type: 1
                            IPv4: 209.165.201.26
                            IPv6_Prefix: 0
                        FQ_TEID:
                            PgwCntrl:
                                IFace: 7
                                TEID: 13210
                                IPv4: 209.165.201.18
                            SgwCntrl:
                                IFace: 11
                                TEID: 301990215
                                IPv4: 209.165.201.19
                        Bearer_Context_List:
                            NumBearerCtxt: 1
                            PbBearerCxt:
                                PbBearerCxt[0]:
```

```
                                        BearerCtxType: 0
                                        EBI: 5
                                        Cause:
                                            Cause_Value: 16
                                            PCE: false
                                            BCE: false
                                            OrigInd: false
                                        Fqteid:
                                            PgwData:
                                                IFace: 5
                                                TEID: 13211
                                                IPv4: 209.165.201.18
                                            SgwData:
                                                IFace: 1
                                                TEID: 25271
                                                IPv4: 209.165.200.226
                                        ChrgId:
                                            Value: 303174163
                                APN_Restriction:
                                    Value: 1

------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.570132
      Message: S11 Create Session Response
      Description: S11 Create Session Response Message
      Source: 209.165.201.19
      Destination: 209.165.201.20
      PAYLOAD:
        S11 Create Session Response:
            S11 Create Session Response:
                Version: 2
                Pflag: false
                TEIDflag: true
                MsgPriority: false
                MsgLength: 120
                TIED: 25269
                Seq: 5842
                MsgTypeId: 33
                MsgPriorityValue: 0
                Peer_IPv4_Flag: false
                Peer_IPv6_Flag: false
                MetaData: From:209.165.201.19:2123->To:209.165.201.20:2123
                Seid: 0
                Rseid: 0
                Cmnid: 0
                MsgType:
                    Create_Session_Response:
                        Cause:
                            Cause_Value: 16
                            PCE: false
                            BCE: false
                            OrigInd: false
                        Recovery:
                            Value: 0
                        AMBR:  UL: 10 kbps, DL: 20 kbps
                        PAA:
                            PDN_Type: 1
                            IPv4: 209.165.201.26
                            IPv6_Prefix: 0
                        FQ_TEID:
                            PgwCntrl:
                                IFace: 7
```

```
                                    TEID: 13210
                                    IPv4: 209.165.201.18
                                SgwCntrl:
                                    IFace: 11
                                    TEID: 301990215
                                    IPv4: 209.165.201.19
                            Bearer_Context_List:
                                NumBearerCtxt: 1
                                PbBearerCxt:
                                    PbBearerCxt[0]:
                                        BearerCtxType: 0
                                        EBI: 5
                                        Cause:
                                            Cause_Value: 16
                                            PCE: false
                                            BCE: false
                                            OrigInd: false
                                        Fqteid:
                                            PgwData:
                                                IFace: 5
                                                TEID: 13211
                                                IPv4: 209.165.201.18
                                            SgwData:
                                                IFace: 1
                                                TEID: 25271
                                                IPv4: 209.165.200.226
                                        ChrgId:
                                            Value: 303174163
                            APN_Restriction:
                                Value: 1

------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.652708
      Message: Sx Session Modification Request
      Description: Sx Session Modification Request Message from SGWC to SGWU
      Source: 209.165.201.19 (SGW.protocol.DC.Local.0)
      Destination: 209.165.201.20 (SGW.udp-proxy.DC.Local.0)
      PAYLOAD:
        Sx Session Modification Request:
            Sx Session Modification Request:
                UpdatePdr:
                    UpdatePdr[0]:
                        PdrId: 2
                        OuterHdrRem: 0
                        Precedence: 0
                        Pdi:
                            SrcIf: ACCESS
                            UeIp:
                                Src: false
                                Dst: false
                                IPv4Addr: 209.165.201.30
                            TEndpointId: 0
                            Valid: false
                        Qfi: 0
                UpdateFar:
                    UpdateFar[0]:
                        FarId: 1
                        ApplyAction:
                            Drop: false
                            Frwd: true
                            Buff: false
                            Nocp: false
```

```
                Dupl: false
                Valid: true
            UpdateFwdParams:
                DestIf: ACCESS
                RedirectInfo:
                    AddrType: 0
                    Valid: false
                OuterHdr:
                    OuterHdrDesc: 256
                    Teid: 25272
                    IPv4Address: 209.165.201.20
                    Port: 0
                    Valid: true
                TEndptId: 0
                OuterPktTos: 0
                InnerPktTos: 0
                TosOpt:
                    CopyInner: false
                    CopyOuter: false
                SendTos: 0
                PfcpSmFlags:
                    Drobu: false
                    Qaurr: false
                    Sndem: false
                    Valid: false
                Valid: true
                NextHopId: 0
            UpdateDuplParams:
                DestIf: ACCESS
                OuterHdr:
                    OuterHdrDesc: 0
                    Teid: 0
                    IPv4Address: 209.165.201.30
                    Port: 0
                    Valid: false
                InterceptInfo:
                    InterceptId: 0
                    ChargingId: 0
                    SmfLiNodeId:
                        IpDesc: 0
                        IPv4Address: 209.165.201.30
                        Valid: false
                    PduSessionId: 0
                    Valid: false
                Valid: false
            BarId: 0
    UplaneInacTimer: 0
    MetaData: From:209.165.201.19:13486->To:209.165.201.20:8805
    Supi:
    Seid: 1297038098512740679
    Seqno: 4245
    Version: 0
    MsgPriority: false
    MsgPriorityVal: 0
    Cmnid: 0
    Rseid: 10002
    IntfType: 0
    HdrLen: 0
    MsgLen: 0
    PfcpSmFlags:
        Drobu: false
        Qaurr: false
        Sndem: false
        Valid: false
```

```
                    UserIDInfo:
                        Valid: false
                    XHeaderInfo:
                        RatType:
                        Valid: false
                    CfPolicyId:
                        PolicyId: 0
                        Valid: false
                    GyStatus:
                        Valid: false
                        Value: false
                    ChargingDisabled:
                        Valid: false
                        Value: false
                    QueryInterface:
                        Valid: false
                        OfflineUrr: false
                        OnlineUrr: false
                        RadiusUrr: false
                        BearerUrr: false
                        SessUrr: false

------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.713891
      Message: Sx Session Modification Response
      Description: Sx Session Modification Response Message from SGWU to SGWC
      Source: 209.165.201.20 (SGW.udp-proxy.DC.Local.0)
      Destination: 209.165.201.19 (SGW.protocol.DC.Local.0)
      PAYLOAD:
        Sx Session Modification Response:
            Sx Session Modification Response:
                Cause: 1
                OffendingIe: 0
                LoadControlInfo:
                    SeqNum: 0
                    Metric: 0
                    Valid: false
                OverloadControlInfo:
                    SeqNum: 0
                    Metric: 0
                    Ociflag: 0
                    Valid: false
                MetaData: From:209.165.201.20:8805->To:209.165.201.19:13486
                Supi:
                Seid: 1297038098512740679
                Seqno: 4245
                Version: 1
                MsgPriority: false
                MsgPriorityVal: 0
                Cmnid: 0
                Rseid: 0
                IntfType: 0
                HdrLen: 17
                MsgLen: 0

------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.640179
      Message: S11 Modify Bearer Request
      Description: S11 Modify Bearer Request Message
      Source: 209.165.201.20
```

```
Destination: 209.165.201.19
PAYLOAD:
  S11 Modify Bearer Request:
      S11 Modify Bearer Request:
          Version: 2
          Pflag: false
          TEIDflag: true
          MsgPriority: false
          MsgLength: 58
          TIED: 301990215
          Seq: 5843
          MsgTypeId: 34
          MsgPriorityValue: 0
          Peer_IPv4_Flag: false
          Peer_IPv6_Flag: false
          MetaData: From:209.165.201.20:2123->To:209.165.201.19:2123
          Seid: 0
          Rseid: 0
          Cmnid: 0
          MsgType:
              Modify_Bearer_Request:
                  RAT_Type:
                      Value: 6
                  Indication:
                      DAF: false
                      DTF: false
                      HI: false
                      DFI: false
                      OI: false
                      ISRSI: false
                      ISRAI: false
                      SGWCI: false
                      SQCI: false
                      UIMSI: false
                      CFSI: false
                      CRSI: false
                      P: false
                      PT: false
                      SI: false
                      MSV: false
                      RetLoc: false
                      PBIC: false
                      SRNI: false
                      S6AF: false
                      S4AF: false
                      MBMDT: false
                      ISRAU: false
                      CCRSI: false
                      CPRAI: false
                      ARRL: false
                      PPOF: false
                      PPON_PPEI: false
                      PPSI: false
                      CSFBI: false
                      CLII: false
                      CPSR: false
                      NSI: false
                      UASI: false
                      DTCI: false
                      BDWI: false
                      PSCI: false
                      PCRI: false
                      AOSI: false
                      AOPI: false
```

```
                                    ROAAI: false
                                    EPCOSI: false
                                    CPOPCI: false
                                    PMTSMI: false
                                    S11TF: false
                                    PNSI: false
                                    UNACCSI: false
                                    WPMSI: false
                                    5GSIWK: false
                                    EEVRSI: false
                                    LTEMUI: false
                                    LTEMPI: false
                                    ENBCRSI: false
                                    TSPCMI: false
                                    PGBK: false
                                    PCPSI: false
                                    PCP: false
                                    PCPU: false
                                    N26_5GS: false
                                    RI_5GCN: false
                                    RS_5GCN: false
                                FQ_TEID:
                                DelayValue:
                                    Value: 0
                                Recovery:
                                    Value: 100
                                Bearer_Context_List:
                                    NumBearerCtxt: 1
                                    PbBearerCxt:
                                        PbBearerCxt[0]:
                                            BearerCtxType: 0
                                            EBI: 5
                                            Fqteid:
                                                ENbData:
                                                    IFace: 0
                                                    TEID: 25272
                                                    IPv4: 209.165.201.20

-------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.729811
      Message: S11 Modify Bearer Response
      Description: S11 Modify Bearer Response Message
      Source: 209.165.201.19
      Destination: 209.165.201.20
      PAYLOAD:
        S11 Modify Bearer Response:
            S11 Modify Bearer Response:
                Version: 2
                Pflag: false
                TEIDflag: true
                MsgPriority: false
                MsgLength: 42
                TIED: 25269
                Seq: 5843
                MsgTypeId: 35
                MsgPriorityValue: 0
                Peer_IPv4_Flag: false
                Peer_IPv6_Flag: false
                MetaData: From:209.165.201.19:2123->To:209.165.201.20:2123
                Seid: 0
                Rseid: 0
                Cmnid: 0
```

```
                    MsgType:
                        Modify_Bearer_Response:
                            Cause:
                                Cause_Value: 16
                                PCE: false
                                BCE: false
                                OrigInd: false
                            Bearer_Context_List:
                                NumBearerCtxt: 1
                                PbBearerCxt:
                                    PbBearerCxt[0]:
                                        BearerCtxType: 0
                                        EBI: 5
                                        Cause:
                                            Cause_Value: 16
                                            PCE: false
                                            BCE: false
                                            OrigInd: false
                                        Fqteid:
                                            SgwData:
                                                IFace: 1
                                                TEID: 25271
                                                IPv4: 209.165.200.226

------------------------------------------------

      Subscriber Id: imsi-123456789012348
      Timestamp: 2021/08/15 12:39:25.730371
      Message: S11 Modify Bearer Response
      Description: S11 Modify Bearer Response Message
      Source: 209.165.201.19
      Destination: 209.165.201.20
      PAYLOAD:
        S11 Modify Bearer Response:
            S11 Modify Bearer Response:
                Version: 2
                Pflag: false
                TEIDflag: true
                MsgPriority: false
                MsgLength: 42
                TIED: 25269
                Seq: 5843
                MsgTypeId: 35
                MsgPriorityValue: 0
                Peer_IPv4_Flag: false
                Peer_IPv6_Flag: false
                MetaData: From:209.165.201.19:2123->To:209.165.201.20:2123
                Seid: 0
                Rseid: 0
                Cmnid: 0
                MsgType:
                    Modify_Bearer_Response:
                        Cause:
                            Cause_Value: 16
                            PCE: false
                            BCE: false
                            OrigInd: false
                        Bearer_Context_List:
                            NumBearerCtxt: 1
                            PbBearerCxt:
                                PbBearerCxt[0]:
                                    BearerCtxType: 0
                                    EBI: 5
                                    Cause:
```

```
                                        Cause_Value: 16
                                        PCE: false
                                        BCE: false
                                        OrigInd: false
                                    Fqteid:
                                        SgwData:
                                            IFace: 1
                                            TEID: 25271
                                            IPv4: 209.165.200.226

-----------------------------------------------

command terminated with exit code 124
  % Total    % Received % Xferd  Average Speed   Time
Time     Time  Current
                                 Dload  Upload   Total
Spent    Left  Speed
100   222  100    35  100   187   4375  23375
--:--:-- --:--:-- --:--:-- 27750
Stop Response Disabled mon_sub as part of timeout for
Cmd: --header Content-type:application/json --request
POST --data
{"commandname":"mon_sub","parameters":{"supi":"imsi-*","duration":300,
"enableTxnLog":false,"enableInternalMsg":false,"action":"stop","namespace":
"sgw","nf-service":"none","grInstance":0}}
http://oam-pod:8879/commands
```

Configuring the Monitor Protocol

To configure this feature, use the following configuration:

```
exec
   monitor protocol interface interface
      capture-duration capture_duration
      pcap [ Yes | No ]
      gr-instance gr_instance
      end
```

NOTES:

- **monitor protocol interface** *interface*—Specify the interface on which PCAP is captured. For example, sbi, pfcp, gtpu, gtpc, gtp, and radius.
- **capture-duration** *capture_duration*—Specify the duration in seconds during which PCAP is captured. The default value is 300 seconds.
- **pcap [Yes | No]** —Configures the PCAP file generation. By default, the pcap feature is disabled.
- **gr-instance** *gr_instance*—Specify the GR instance that the cnSGW-C monitors the subscriber for.

Note If the GTP endpoint IPs are the same on S5e and S11 interfaces, the protocol output is inconsistent and displays S11 for the S5 interface on which the message is received. The following is a sample of an endpoint configuration:

```
instance instance-id 1 endpoint gtp replicas 3 interface s5e vip-ip 209.165.201.22
instance instance-id 1 endpoint gtp replicas 3 interface s11 vip-ip 209.165.201.22
```

Configuration Example

The following is an example configuration.

```
monitor protocol interface pfcp,gtpc capture-duration 100 pcap yes
```

Sample Output

The following is a sample output.

```
monitor protocol interface pcfp,gtpc capture-duration 100 pcap yes
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   231  100   101  100   130   6733   8666 --:--:-- --:--:-- --:--:-- 15400
Command: --header Content-type:application/json --request POST --data
{"commandname":"mon_pro","parameters":{"interface":"pcfp, gtpc","duration"
:100,"action":"start","enable_pcap": true,"grInstance":0}} http://oam-pod:8879/commands
Result start mon_pro, fileName -> logs/monprologs/sessintfname_pcfp,
gtpc_at_2021-06-06T06:15:39.005414271.txt
Starting to tail the monpro messages from file: logs/monprologs/sessintfname_pcfp,
gtpc_at_2021-06-06T06:15:39.005414271.txt
Defaulting container name to oam-pod.
Use 'kubectl describe pod/oam-pod-0 -n cn' to see all of the containers in this pod.
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:00.373101 +0000 UTC CaptureLength:59
Length:59 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002dcc0e4000401
155510a01029c0a01029b3a9a084b0019196348630000d000008d0
020175004900010005

Packet Dump:
 -- FULL PACKET DATA (59 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 2d cc 0e 40 00 40 11  55 51 0a 01 02 9c 0a 01  |.-..@.@.UQ......|
00000020  02 9b 3a 9a 08 4b 00 19  19 63 48 63 00 0d 00 00  |..:..K...cHc....|
00000030  08 d0 02 01 75 00 49 00  01 00 05                 |....u.I....|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..45..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..25..] Version=4 IHL=5 TOS=40 Length=45
Id=52238 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=21841 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 2d cc 0e 40 00  40 11 55 51 0a 01 02 9c  |E(.-..@.@.UQ....|
00000010  0a 01 02 9b                                      |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..17..] SrcPort=15002 DstPort=2123(gtp-control)
Length=25 Checksum=6499}
00000000  3a 9a 08 4b 00 19 19 63                          |:..K...c|
--- Layer 4 ---
Payload 17 byte(s)
00000000  48 63 00 0d 00 00 08 d0  02 01 75 00 49 00 01 00  |Hc........u.I...|
00000010  05                                               |.|


------------------------------------------------------------------------
```

```
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15000 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:00.373236 +0000 UTC CaptureLength:72
Length:72 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528003a9af
c4000401186570a01029c0a01029a3a98084b0026196f 4824001a0000074a
00014f00490001000557000900865100000970a01029c

Packet Dump:
 -- FULL PACKET DATA (72 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 3a 9a fc 40 00 40 11  86 57 0a 01 02 9c 0a 01  |.:..@.@..W......|
00000020  02 9a 3a 98 08 4b 00 26  19 6f 48 24 00 1a 00 00  |..:..K.&.oH$....|
00000030  07 4a 00 01 4f 00 49 00  01 00 05 57 00 09 00 86  |.J..O.I....W....|
00000040  51 00 00 97 0a 01 02 9c                           |Q.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..58..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..38..] Version=4 IHL=5 TOS=40 Length=58
Id=39676 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=34391 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 3a 9a fc 40 00  40 11 86 57 0a 01 02 9c  |E(.:..@.@..W....|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..30..] SrcPort=15000(hydap) DstPort=2123(gtp-control)
Length=38 Checksum=6511}
00000000  3a 98 08 4b 00 26 19 6f                           |:..K.&.o|
--- Layer 4 ---
Payload 30 byte(s)
00000000  48 24 00 1a 00 00 07 4a  00 01 4f 00 49 00 01 00  |H$.....J..O.I...|
00000010  05 57 00 09 00 86 51 00  00 97 0a 01 02 9c        |.W....Q.......|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15000 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:02.462216 +0000 UTC CaptureLength:72
Length:72 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528003a9cdc4000401184770a01029c0a
01029a3a98084b0026196f4824001a0000074a00014f00490001000557000900865100000970a01029c

Packet Dump:
 -- FULL PACKET DATA (72 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 3a 9c dc 40 00 40 11  84 77 0a 01 02 9c 0a 01  |.:..@.@..w......|
00000020  02 9a 3a 98 08 4b 00 26  19 6f 48 24 00 1a 00 00  |..:..K.&.oH$....|
00000030  07 4a 00 01 4f 00 49 00  01 00 05 57 00 09 00 86  |.J..O.I....W....|
00000040  51 00 00 97 0a 01 02 9c                           |Q.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..58..] SrcMAC=00:50:56:9c:09:59
```

```
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..38..] Version=4 IHL=5 TOS=40 Length=58
Id=40156 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=33911 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 3a 9c dc 40 00  40 11 84 77 0a 01 02 9c  |E(.:..@.@..w....|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..30..] SrcPort=15000(hydap) DstPort=2123(gtp-control)
Length=38 Checksum=6511}
00000000  3a 98 08 4b 00 26 19 6f                          |:..K.&.o|
--- Layer 4 ---
Payload 30 byte(s)
00000000  48 24 00 1a 00 00 07 4a  00 01 4f 00 49 00 01 00  |H$.....J..O.I...|
00000010  05 57 00 09 00 86 51 00  00 97 0a 01 02 9c        |.W....Q.......|


------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15000 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:04.563024 +0000 UTC CaptureLength:72
Length:72 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
000400010006005056c09590800452800 3a9e4740004011830c0a01029c0a01029a3a98084b
0026196f4824001a0000074a00014f00490001000557000900865100009 70a01029c

Packet Dump:
 -- FULL PACKET DATA (72 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 3a 9e 47 40 00 40 11  83 0c 0a 01 02 9c 0a 01  |.:.G@.@.........|
00000020  02 9a 3a 98 08 4b 00 26  19 6f 48 24 00 1a 00 00  |..:..K.&.oH$....|
00000030  07 4a 00 01 4f 00 49 00  01 00 05 57 00 09 00 86  |.J..O.I....W....|
00000040  51 00 00 97 0a 01 02 9c                          |Q.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..58..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..38..] Version=4 IHL=5 TOS=40 Length=58
Id=40519 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=33548 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 3a 9e 47 40 00  40 11 83 0c 0a 01 02 9c  |E(.:.G@.@.......|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..30..] SrcPort=15000(hydap)
DstPort=2123(gtp-control) Length=38 Checksum=6511}
00000000  3a 98 08 4b 00 26 19 6f                          |:..K.&.o|
--- Layer 4 ---
Payload 30 byte(s)
00000000  48 24 00 1a 00 00 07 4a  00 01 4f 00 49 00 01 00  |H$.....J..O.I...|
00000010  05 57 00 09 00 86 51 00  00 97 0a 01 02 9c        |.W....Q.......|


------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
```

```
from 209.165.200.229:15000 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:06.662105 +0000 UTC CaptureLength:72
Length:72 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528003a9eb440004011829f0a01029c0a01029a3a
98084b0026196f4824001a0000074a00014f00490001000557000900865100000970a01029c

Packet Dump:
 -- FULL PACKET DATA (72 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 3a 9e b4 40 00 40 11  82 9f 0a 01 02 9c 0a 01  |.:..@.@.........|
00000020  02 9a 3a 98 08 4b 00 26  19 6f 48 24 00 1a 00 00  |..:..K.&.oH$....|
00000030  07 4a 00 01 4f 00 49 00  01 00 05 57 00 09 00 86  |.J..O.I....W....|
00000040  51 00 00 97 0a 01 02 9c                          |Q.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..58..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..38..] Version=4 IHL=5 TOS=40 Length=58
Id=40628 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=33439 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 3a 9e b4 40 00  40 11 82 9f 0a 01 02 9c  |E(.:..@.@.......|
00000010  0a 01 02 9a                                      |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..30..] SrcPort=15000(hydap)
DstPort=2123(gtp-control) Length=38 Checksum=6511}
00000000  3a 98 08 4b 00 26 19 6f                          |:..K.&.o|
--- Layer 4 ---
Payload 30 byte(s)
00000000  48 24 00 1a 00 00 07 4a  00 01 4f 00 49 00 01 00  |H$.....J..O.I...|
00000010  05 57 00 09 00 86 51 00  00 97 0a 01 02 9c        |.W....Q.......|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:07.385688 +0000 UTC CaptureLength:59
Length:59 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002dd2dd400040114e820a0102
9c0a01029b3a9a084b001919634863000d00000 8d0020175004900010005

Packet Dump:
 -- FULL PACKET DATA (59 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 2d d2 dd 40 00 40 11  4e 82 0a 01 02 9c 0a 01  |.-..@.@.N.......|
00000020  02 9b 3a 9a 08 4b 00 19  19 63 48 63 00 0d 00 00  |..:..K...cHc....|
00000030  08 d0 02 01 75 00 49 00  01 00 05                 |....u.I....|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..45..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..25..] Version=4 IHL=5 TOS=40 Length=45
Id=53981 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=20098 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
```

```
00000000  45 28 00 2d d2 dd 40 00  40 11 4e 82 0a 01 02 9c  |E(.-..@.@.N.....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..17..] SrcPort=15002 DstPort=2123(gtp-control)
Length=25 Checksum=6499}
00000000  3a 9a 08 4b 00 19 19 63                          |:..K...c|
--- Layer 4 ---
Payload 17 byte(s)
00000000  48 63 00 0d 00 00 08 d0  02 01 75 00 49 00 01 00  |Hc........u.I...|
00000010  05                                                |.|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15001 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:13.078691 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002 9a11540004011804f0a01029c0
a01029a3a99084b0015195e400100090101ab000300010000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 a1 15 40 00 40 11  80 4f 0a 01 02 9c 0a 01  |.)..@.@..O......|
00000020  02 9a 3a 99 08 4b 00 15  19 5e 40 01 00 09 01 01  |..:..K...^@.....|
00000030  ab 00 03 00 01 00 00                             |.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=41237 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=32847 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 29 a1 15 40 00  40 11 80 4f 0a 01 02 9c  |E(.)..@.@..O....|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15001 DstPort=2123(gtp-control)
Length=21 Checksum=6494}
00000000  3a 99 08 4b 00 15 19 5e                          |:..K...^|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 01 01 ab 00  03 00 01 00 00           |@............|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15000 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:13.084971 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002 9d5a7400040114bbc0a01029c0a01029b3
a98084b0015195f400100090000df000300010000
```

```
Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 d5 a7 40 00 40 11  4b bc 0a 01 02 9c 0a 01  |.)..@.@.K.......|
00000020  02 9b 3a 98 08 4b 00 15  19 5f 40 01 00 09 00 00  |..:..K..._@.....|
00000030  df 00 03 00 01 00 00                              |.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=54695 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=19388 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 29 d5 a7 40 00  40 11 4b bc 0a 01 02 9c  |E(.)..@.@.K.....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15000(hydap) DstPort=2123(gtp-control)
Length=21 Checksum=6495}
00000000  3a 98 08 4b 00 15 19 5f                           |:..K..._|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 00 00 df 00  03 00 01 00 00           |@............|


--------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:14.485021 +0000 UTC CaptureLength:59
Length:59 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
000400010006005056c09590800452800 2dd679400040114ae60a01029c0a01029b3
a9a084b001919634863000d000008d002017500490001 0005

Packet Dump:
 -- FULL PACKET DATA (59 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 2d d6 79 40 00 40 11  4a e6 0a 01 02 9c 0a 01  |.-.y@.@.J.......|
00000020  02 9b 3a 9a 08 4b 00 19  19 63 48 63 00 0d 00 00  |..:..K...cHc....|
00000030  08 d0 02 01 75 00 49 00  01 00 05                 |....u.I....|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..45..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..25..] Version=4 IHL=5 TOS=40 Length=45
Id=54905 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=19174 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 2d d6 79 40 00  40 11 4a e6 0a 01 02 9c  |E(.-.y@.@.J.....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..17..] SrcPort=15002 DstPort=2123(gtp-control)
Length=25 Checksum=6499}
00000000  3a 9a 08 4b 00 19 19 63                           |:..K...c|
--- Layer 4 ---
Payload 17 byte(s)
00000000  48 63 00 0d 00 00 08 d0  02 01 75 00 49 00 01 00  |Hc........u.I...|
00000010  05                                                |.|
```

```
------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:18.078521 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002 9a1bb400040117fa90a01029c0a01029a3
a9a084b0015195e400100090201c9000300010000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 a1 bb 40 00 40 11  7f a9 0a 01 02 9c 0a 01  |.)..@.@.........|
00000020  02 9a 3a 9a 08 4b 00 15  19 5e 40 01 00 09 02 01  |..:..K...^@.....|
00000030  c9 00 03 00 01 00 00                              |.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=41403 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=32681 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 29 a1 bb 40 00  40 11 7f a9 0a 01 02 9c  |E(.)..@.@.......|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15002 DstPort=2123(gtp-control)
Length=21 Checksum=6494}
00000000  3a 9a 08 4b 00 15 19 5e                           |:..K...^|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 02 01 c9 00  03 00 01 00 00           |@............|


------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15000 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:18.089434 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002 9d8cf4000401148940a0102
9c0a01029b3a98084b0015195f400100090000e0000300010000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 d8 cf 40 00 40 11  48 94 0a 01 02 9c 0a 01  |.)..@.@.H.......|
00000020  02 9b 3a 98 08 4b 00 15  19 5f 40 01 00 09 00 00  |..:..K..._@.....|
00000030  e0 00 03 00 01 00 00                              |.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
```

```
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=55503 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=18580 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 29 d8 cf 40 00  40 11 48 94 0a 01 02 9c  |E(.)..@.@.H.....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15000(hydap) DstPort=2123(gtp-control)
Length=21 Checksum=6495}
00000000  3a 98 08 4b 00 15 19 5f                           |:..K..._|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 00 00 e0 00  03 00 01 00 00           |@............|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:21.584999 +0000 UTC CaptureLength:59
Length:59 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c095908004528002ddb124000401146
4d0a01029c0a01029b3a9a084b001919634863000d00 0008d0020175004900010005

Packet Dump:
 -- FULL PACKET DATA (59 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 2d db 12 40 00 40 11  46 4d 0a 01 02 9c 0a 01  |.-..@.@.FM......|
00000020  02 9b 3a 9a 08 4b 00 19  19 63 48 63 00 0d 00 00  |..:..K...cHc....|
00000030  08 d0 02 01 75 00 49 00  01 00 05                 |....u.I....|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..45..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..25..] Version=4 IHL=5 TOS=40 Length=45
Id=56082 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=17997 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 2d db 12 40 00  40 11 46 4d 0a 01 02 9c  |E(.-..@.@.FM....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..17..] SrcPort=15002 DstPort=2123(gtp-control)
Length=25 Checksum=6499}
00000000  3a 9a 08 4b 00 19 19 63                           |:..K...c|
--- Layer 4 ---
Payload 17 byte(s)
00000000  48 63 00 0d 00 00 08 d0  02 01 75 00 49 00 01 00  |Hc........u.I...|
00000010  05                                                |.|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15001 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:23.07887 +0000 UTC CaptureLength:55
```

```
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c0959080045280029a2c0400040117
ea40a01029c0a01029a3a99084b0015195e400100090101ac000300010000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 a2 c0 40 00 40 11  7e a4 0a 01 02 9c 0a 01  |.)..@.@.~.......|
00000020  02 9a 3a 99 08 4b 00 15  19 5e 40 01 00 09 01 01  |..:..K...^@.....|
00000030  ac 00 03 00 01 00 00                              |.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=41664 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=32420 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 29 a2 c0 40 00  40 11 7e a4 0a 01 02 9c  |E(.)..@.@.~.....|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15001 DstPort=2123(gtp-control)
Length=21 Checksum=6494}
00000000  3a 99 08 4b 00 15 19 5e                          |:..K...^|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 01 01 ac 00  03 00 01 00 00           |@............|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:23.086144 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c0959080045280029dbe740004011457c0a01029c0a01029b3
a9a084b0015195f400100090201760003000100000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 db e7 40 00 40 11  45 7c 0a 01 02 9c 0a 01  |.)..@.@.E|......|
00000020  02 9b 3a 9a 08 4b 00 15  19 5f 40 01 00 09 02 01  |..:..K..._@.....|
00000030  76 00 03 00 01 00 00                              |v......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=56295 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=17788 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 29 db e7 40 00  40 11 45 7c 0a 01 02 9c  |E(.)..@.@.E|....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15002 DstPort=2123(gtp-control)
Length=21 Checksum=6495}
```

```
00000000  3a 9a 08 4b 00 15 19 5f                           |:..K..._|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 02 01 76 00  03 00 01 00 00           |@.....v.....|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.227:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:28.079013 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c0959080045280029a336400040117e2e0a01029c0
a01029a3a9a084b0015195e400100090201ca000300010000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 a3 36 40 00 40 11  7e 2e 0a 01 02 9c 0a 01  |.).6@.@.~.......|
00000020  02 9a 3a 9a 08 4b 00 15  19 5e 40 01 00 09 02 01  |..:..K...^@.....|
00000030  ca 00 03 00 01 00 00                              |.......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=41782 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=32302 SrcIP=209.165.200.229 DstIP=209.165.200.227 Options=[] Padding=[]}
00000000  45 28 00 29 a3 36 40 00  40 11 7e 2e 0a 01 02 9c  |E(.).6@.@.~.....|
00000010  0a 01 02 9a                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15002 DstPort=2123(gtp-control)
Length=21 Checksum=6494}
00000000  3a 9a 08 4b 00 15 19 5e                           |:..K...^|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 02 01 ca 00  03 00 01 00 00           |@.....|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:28.086562 +0000 UTC CaptureLength:55
Length:55 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
0004000100060050569c0959080045280029dfb14000401141b20a01029
c0a01029b3a9a084b0015195f400100090201770003000100000

Packet Dump:
 -- FULL PACKET DATA (55 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 29 df b1 40 00 40 11  41 b2 0a 01 02 9c 0a 01  |.)..@.@.A.......|
00000020  02 9b 3a 9a 08 4b 00 15  19 5f 40 01 00 09 02 01  |..:..K..._@.....|
```

```
00000030  77 00 03 00 01 00 00                              |w......|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..41..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..21..] Version=4 IHL=5 TOS=40 Length=41
Id=57265 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=16818 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 29 df b1 40 00  40 11 41 b2 0a 01 02 9c  |E(.)..@.@.A.....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..13..] SrcPort=15002 DstPort=2123(gtp-control)
Length=21 Checksum=6495}
00000000  3a 9a 08 4b 00 15 19 5f                           |:..K..._|
--- Layer 4 ---
Payload 13 byte(s)
00000000  40 01 00 09 02 01 77 00  03 00 01 00 00           |@.....w......|


------------------------------------------------------------------------
InterfaceName = gtpc |  InterfaceIP = 209.165.200.229 | Filter = (tcp or udp or sctp)
and (port 2123 or (host 209.165.200.227 and port 2123)
or (host 209.165.201.11 and port 2123) or (host 209.165.200.228 and port 2123))
<<<<OUTBOUND
from 209.165.200.229:15002 to 209.165.200.228:2123
Protocol: UDP | Sequence Number: 0
Packet Metadata: {Timestamp:2021-06-06 06:16:28.685112 +0000 UTC CaptureLength:59
Length:59 InterfaceIndex:0 AncillaryData:[]}

Packet Raw Bytes:
000400010006005056c09590800452800 2de03240004011412d0a01029c0a01029b3a9a
084b001919634863000d000008d00201750049000 10005

Packet Dump:
 -- FULL PACKET DATA (59 bytes) ------------------------------------
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00 45 28  |.......PV..Y..E(|
00000010  00 2d e0 32 40 00 40 11  41 2d 0a 01 02 9c 0a 01  |.-.2@.@.A-......|
00000020  02 9b 3a 9a 08 4b 00 19  19 63 48 63 00 0d 00 00  |..:..K...cHc....|
00000030  08 d0 02 01 75 00 49 00  01 00 05                 |....u.I....|
--- Layer 1 ---
Ethernet        {Contents=[..14..] Payload=[..45..] SrcMAC=00:50:56:9c:09:59
DstMAC=00:04:00:01:00:06 EthernetType=IPv4 Length=0}
00000000  00 04 00 01 00 06 00 50  56 9c 09 59 08 00        |.......PV..Y..|
--- Layer 2 ---
IPv4    {Contents=[..20..] Payload=[..25..] Version=4 IHL=5 TOS=40 Length=45
Id=57394 Flags=DF FragOffset=0 TTL=64 Protocol=UDP
Checksum=16685 SrcIP=209.165.200.229 DstIP=209.165.200.228 Options=[] Padding=[]}
00000000  45 28 00 2d e0 32 40 00  40 11 41 2d 0a 01 02 9c  |E(.-.2@.@.A-....|
00000010  0a 01 02 9b                                       |....|
--- Layer 3 ---
UDP     {Contents=[..8..] Payload=[..17..] SrcPort=15002 DstPort=2123(gtp-control)
Length=25 Checksum=6499}
00000000  3a 9a 08 4b 00 19 19 63                           |:..K...c|
--- Layer 4 ---
Payload 17 byte(s)
00000000  48 63 00 0d 00 00 08 d0  02 01 75 00 49 00 01 00  |Hc........u.I...|
00000010  05                                                |.|


------------------------------------------------------------------------
command terminated with exit code 124
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
```

```
100   172  100    43  100   129   8600  25800 --:--:-- --:--:-- --:--:-- 34400
Stop Response Already disabled mon_pro as part of timeout for Cmd: --header
Content-type:application/json --request POST --data
{"commandname":"mon_pro","parameters": {"interface":"pcfp,gtpc","duration":100, "action":
"stop","enable_pcap":true,"grInstance":0}} http://oam-pod:8879/commands
```

Configuring the Transaction Messages

To configure the transaction logs, use the following configuration:

```
config
   logging transaction message [ disable | enable ]
   end
```

NOTES:

- **logging transaction message [disable | enable]**—Configure the messages in transaction logging. When set to enable, the transactional and internal logs are combined. By default, the logs are disabled.

Configuration Example

The following is an example configuration.

```
logging transaction message enable
```

Sample Output

The following is a sample output.

```
/opt/workspace/monsublogs# cat 'sgw.imsi-*_WithTxnLogs_TS_2020-12-22T15:16:58.158964842.txt'

Transaction Log received from Instance: smf.nodemgr.cn.cn.0
************* TRANSACTION: 00088 *************
TRANSACTION SUCCESS:
    Txn Type             : NmgrRersourceMgmtRequest(1025)
    Priority             : 1
    Session Namespace    : sgw(2)
LOG MESSAGES:
    2020/12/22 15:20:07.093 [TRACE] [infra.message_log.core] >>>>>>>>
IPC message
Name: NmgrRersourceMgmtRequest
MessageType: NmgrRersourceMgmtRequest
Key:
--body--
{"supi":"imsi-123456789012348","idReqType":1,"serviceUsertype":2}
    2020/12/22 15:20:07.093 [DEBUG] [nodemgr0.app.Int]
                GetSessionNamespace for txn id: 88, Type: 1025
    2020/12/22 15:20:07.093 [DEBUG] [nodemgr0.app.Int]
                GetSessionNamespace returned namespace : 2
    2020/12/22 15:20:07.093 [INFO] [nodemgr0.app.Int]
                RECEIVED REQUEST <-- NmgrRersourceMgmtRequest
    2020/12/22 15:20:07.093 [TRACE] [infra.message_log.core] <<<<<<<<

*******************************************
Transaction Log received from Instance: smf.protocol.cn.cn.0
************* TRANSACTION: 00050 *************
TRANSACTION SUCCESS:
    Txn Type             : 524(524)
    Priority             : 1
    Session Namespace    : sgw(2)
```

```
INCOMING REQUEST:
      Message: Sx Session Establishment Request
      Description: Sx Session Establishment Request Message from SGWC to SGWU
      Source: 209.165.200.229
      Destination: 209.165.200.228
      PAYLOAD:
        Sx Session Establishment Request:
            Sx Session Establishment Request:
                CreatePdr:
                    CreatePdr[0]:
                        PdrId: 1
                        Precedence: 0
                        Pdi:
                            SrcIf: CORE
                            UeIp:
                                Src: false
                                Dst: false
                                IPv4Addr: 209.165.201.30
                            TEndpointId: 1
                            Valid: true
                        OuterHdrRem: 0
                        FarId:
                            FarId[0]: 1
                        Qfi: 0
                        OuterHdrRemValid: false
                    CreatePdr[1]:
                        PdrId: 2
                        Precedence: 0
                        Pdi:
                            SrcIf: ACCESS
                            UeIp:
                                Src: false
                                Dst: false
                                IPv4Addr: 209.165.201.30
                            TEndpointId: 2
                            Valid: true
                        OuterHdrRem: 0
                        FarId:
                            FarId[0]: 2
                        Qfi: 0
                        OuterHdrRemValid: false
                CreateFar:
                    CreateFar[0]:
                        FarId: 1
                        ApplyAction:
                            Drop: true
                            Frwd: false
                            Buff: false
                            Nocp: false
                            Dupl: false
                            Valid: true
                        FwdParams:
                            DestIf: ACCESS
                            RedirectInfo:
                                AddrType: 0
                                Valid: false
                            OuterHdr:
                                OuterHdrDesc: 0
                                Teid: 0
                                IPv4Address: 209.165.201.30
                                Port: 0
                                Valid: false
                            TEndptId: 2
                            OuterPktTos: 255
```

```
            InnerPktTos: 255
            TosOpt:
                CopyInner: false
                CopyOuter: false
            SendTos: 0
            PfcpSmFlags:
                Drobu: false
                Qaurr: false
                Sndem: false
                Valid: false
            Valid: true
        DuplParams:
            DestIf: ACCESS
            OuterHdr:
                OuterHdrDesc: 0
                Teid: 0
                IPv4Address: 209.165.201.30
                Port: 0
                Valid: false
            InterceptInfo:
                InterceptId: 0
                ChargingId: 0
                SmfLiNodeId:
                    IpDesc: 0
                    IPv4Address: 209.165.201.30
                    Valid: false
                PduSessionId: 0
                Valid: false
            Valid: false
        BarId: 0
    CreateFar[1]:
        FarId: 2
        ApplyAction:
            Drop: true
            Frwd: false
            Buff: false
            Nocp: false
            Dupl: false
            Valid: true
        FwdParams:
            DestIf: CORE
            RedirectInfo:
                AddrType: 0
                Valid: false
            OuterHdr:
                OuterHdrDesc: 0
                Teid: 0
                IPv4Address: 209.165.201.30
                Port: 0
                Valid: false
            TEndptId: 1
            OuterPktTos: 255
            InnerPktTos: 255
            TosOpt:
                CopyInner: false
                CopyOuter: false
            SendTos: 0
            PfcpSmFlags:
                Drobu: false
                Qaurr: false
                Sndem: false
                Valid: false
            Valid: true
        DuplParams:
```

```
                                DestIf: ACCESS
                                OuterHdr:
                                    OuterHdrDesc: 0
                                    Teid: 0
                                    IPv4Address: 209.165.201.30
                                    Port: 0
                                    Valid: false
                                InterceptInfo:
                                    InterceptId: 0
                                    ChargingId: 0
                                    SmfLiNodeId:
                                        IpDesc: 0
                                        IPv4Address: 209.165.201.30
                                        Valid: false
                                    PduSessionId: 0
                                    Valid: false
                                Valid: false
                            BarId: 0
                        CreateTEndpt:
                            CreateTEndpt[0]:
                                EndpointId: 1
                                FTeid:
                                    Teid: 0
                                    IPv4Address: 209.165.201.30
                                    ChooseId: 0
                                BearerLvlInfo:
                                    Valid: 1
                                    Qci: 6
                            CreateTEndpt[1]:
                                EndpointId: 2
                                FTeid:
                                    Teid: 0
                                    IPv4Address: 209.165.201.30
                                    ChooseId: 0
                                BearerLvlInfo:
                                    Valid: 1
                                    Qci: 6
                        PdnType: 0
                        UplaneInacTimer: 0
                        MetaData: From:209.165.200.229:11000->To:209.165.200.228:8805 Seqno:37
                                  Supi: Seid:1224979349111832634 Cmnid:0 Rseid:0 IntfType:0
                        UserIDInfo:
                            Valid: false
                        XHeaderInfo:
                            RatType:
                            Valid: false
                        CfPolicyId:
                            PolicyId: 0
                            Valid: false
                        ChargingDisabled:
                            Valid: false
                            Value: false
                        ChargingParams:
                            Valid: 0
                            GyOfflineChargingEnabled: 0

OUTGOING RESPONSE:
      Message: Sx Session Establishment Response
      Description: Sx Session Establishment Response Message from SGWU to SGWC
      Source: 209.165.200.228
      Destination: 209.165.200.229
      PAYLOAD:
        Sx Session Establishment Response:
            Sx Session Establishment Response:
```

```
                    Cause: 1
                    OffendingIe: 0
                    FSeid:
                        Seid: 10002
                        IPv4Address: 209.165.200.228
                    CreatedTEndpt:
                        CreatedTEndpt[0]:
                            EndpointId: 1
                            FTeid:
                                Teid: 2086
                                IPv4Address: 209.165.201.1
                                ChooseId: 0
                        CreatedTEndpt[1]:
                            EndpointId: 2
                            FTeid:
                                Teid: 2087
                                IPv4Address: 209.165.200.226
                                ChooseId: 0
                    MetaData: From:209.165.200.228:8805->To:209.165.200.229:11000 Seqno:37
                                Supi: Seid:1224979349111832634 Cmnid:0 Rseid:0 IntfType:0
                    LoadControlInfo:
                        SeqNum: 0
                        Metric: 0
                        Valid: false
                    OverloadControlInfo:
                        SeqNum: 0
                        Metric: 0
                        Ociflag: 0
                        Valid: false

LOG MESSAGES:
    2020/12/22 15:20:10.934 [TRACE] [infra.message_log.core] >>>>>>>>

    2020/12/22 15:20:10.934 [DEBUG] [proto_ep.app.Int]
                            GetSessionNamespace for txn id: 50, Type: 524
    2020/12/22 15:20:10.934 [DEBUG]
                            [proto_ep.app.Int] GetSessionNamespace returned namespace : 2
    2020/12/22 15:20:10.934 [DEBUG]
                            [infra.ipc_action.core] BG IPC will be executed after process
continue.
    2020/12/22 15:20:10.934 [DEBUG]
                            [infra.ipc_action.core] BG IPC is executing after process
continue.
    2020/12/22 15:20:10.967 [DEBUG]
                            [infra.transaction.core] Response received for corelation ID
app_protocol-0_37
    2020/12/22 15:20:10.967 [DEBUG]
                            [infra.transaction.core]
                           (1) All pending ipc action finshed, hence waking up transaction

    2020/12/22 15:20:10.967 [TRACE]
                            [infra.message_log.core] <<<<<<<<
```

Accessing the Logs

To access the monitor subscriber logs, navigate to the oam-pod at `/opt/workspace/logs/monsublogs`

Sample Logs

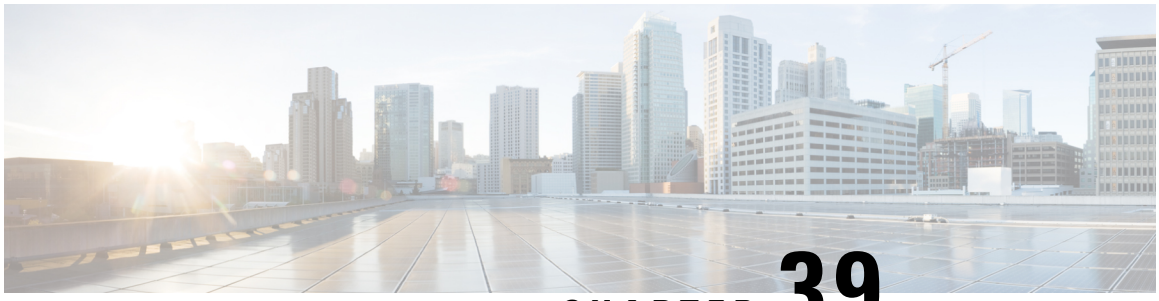
```
root@oam-pod-0:/opt/workspace/logs/monsublogs# ls
none.imsi-123456789_TS_2021-06-05T06:19:12.682444275.txt
sgw.imei-352099001761480_TS_2021-06-05T13:07:41.774214146.txt.sorted
```

```
 none.imsi-123456789_TS_2021-06-05T06:20:39.751939118.txt
sgw.imei-352099001761480_TS_2021-06-05T13:48:51.868279985.txt
 none.imsi-123456789_TS_2021-06-06T06:22:16.015635407.txt
sgw.imei-352099001761480_TS_2021-06-05T13:48:51.868279985.txt.sorted
 sgw.imei-352099001761480_TS_2021-06-04T19:09:24.863985017.txt
sgw.imei-352099001761480_TS_2021-06-05T14:50:09.330635953.txt
 sgw.imei-352099001761480_TS_2021-06-04T19:09:24.863985017.txt.sorted
sgw.imei-352099001761480_TS_2021-06-05T14:50:09.330635953.txt.sorted
 sgw.imei-352099001761480_TS_2021-06-05T08:44:25.889632126.txt
sgw.imei-352099001761480_TS_2021-06-05T17:36:17.238331396.txt
 sgw.imei-352099001761480_TS_2021-06-05T08:44:25.889632126.txt.sorted
sgw.imei-352099001761480_TS_2021-06-05T17:36:17.238331396.txt.sorted
 sgw.imei-352099001761480_TS_2021-06-05T10:26:23.529652777.txt
'sgw.imsi-*_TS_2021-06-05T06:23:19.865508390.txt'
 sgw.imei-352099001761480_TS_2021-06-05T10:26:23.529652777.txt.sorted
'sgw.imsi-*_TS_2021-06-05T06:25:18.219875282.txt'
 sgw.imei-352099001761480_TS_2021-06-05T13:07:41.774214146.txt
```

To access the monitor protocol logs, navigate to the oam-pod at `/opt/workspace/logs/monprologs`.

Sample Logs

```
root@oam-pod-0:/opt/workspace/logs/monprologs# ls
sessintfname_gtpc_at_2021-06-05T06:28:18.310397784.pcap
sessintfname_pcfp,gtpc_at_2021-06-06T06:15:39.005414271.pcap
sessintfname_gtpc_at_2021-06-05T06:28:18.310397784.txt
sessintfname_pcfp,gtpc_at_2021-06-06T06:15:39.005414271.txt
```



CHAPTER 39

Multiple PDN Attach or Detach Procedures

- Feature Summary and Revision History, on page 439
- Feature Description, on page 439
- How it Works, on page 440

Feature Summary and Revision History

Summary Data

Table 176: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 177: Revision History

Revision Details	Release
First introduced.	2020.03.0

Feature Description

cnSGW-C handles the following functionalities:

- UE-requested PDN connection
- UE-requested PDN disconnection

- PGW-initiated PDN disconnection
- Admin-initiated disconnection.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

UE-requested PDN Connection Call Flow

This section describes the UE-requested PDN connection call flow.

Figure 95: UE-requested PDN Connection Call Flow

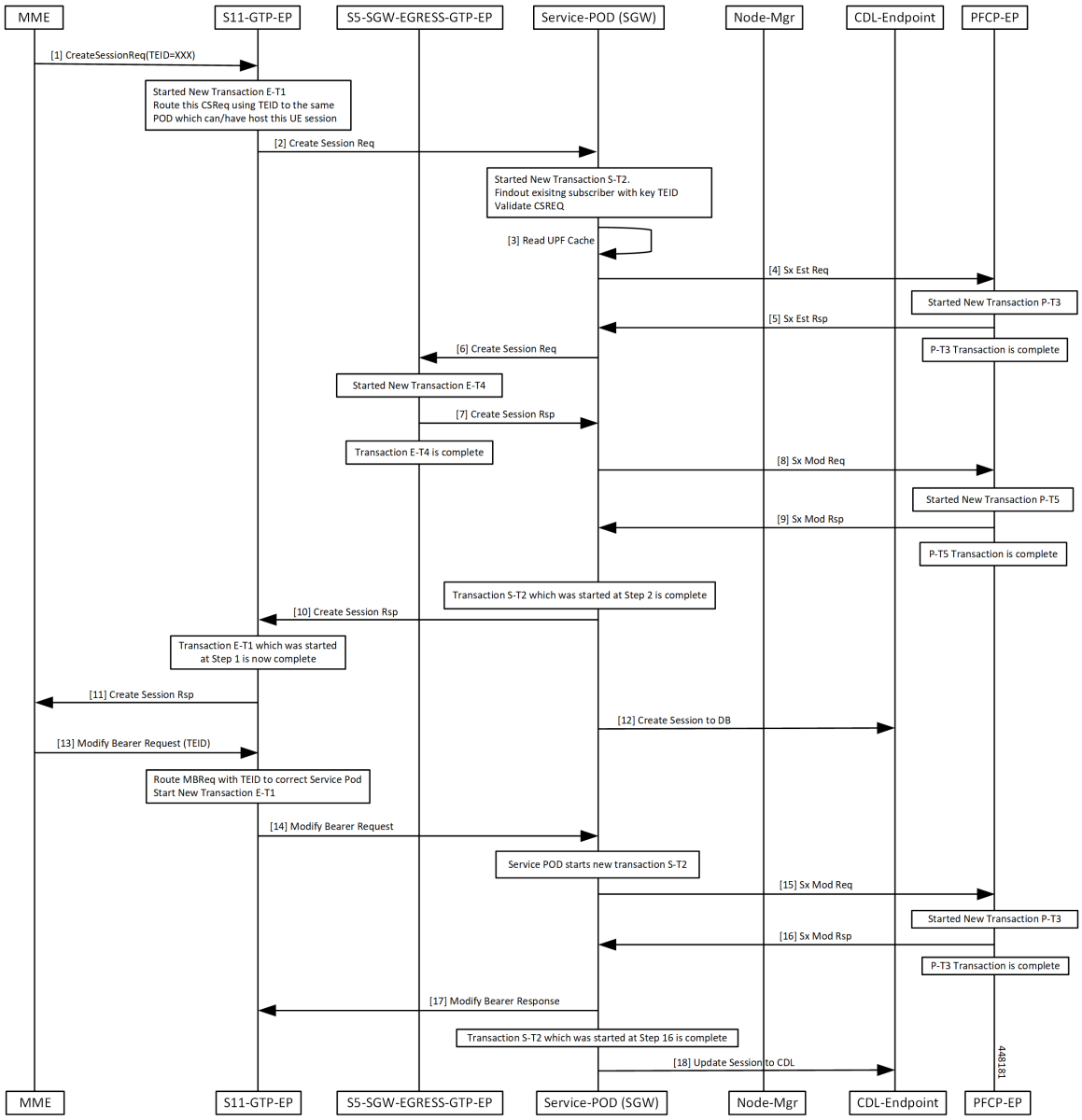


Table 178: UE-requested PDN Connection Call Flow Description

Step	Description
1	The MME sends the Create Session Request for a new PDN connection to the S11-GTP-EP with a nonzero TEID.

Step	Description
2	The E-T1 transaction is started. The S11-GTP-EP decodes the received UDP message and converts the message into the gGRPC message. Based on the IMSI value, the message is forwarded to the respective SGW-service pod which handles the UE session. The SGW-service pod receives the Create Session Request from the S11-GTP-EP.
3	The S-T2 transaction is started. The SGW-service pod finds the subscriber context based on the local ingress TEID, and validates the Create Session Request content and updates the PDN and subscriber information. The SGW-service pod reads the UPF cache to provide the selected UPF.
4	The SGW-service pod sends the Sx Establishment Request to the PFCP-EP.
5	The P-T3 transaction is started. The PFCP-EP sends the Sx Establishment Response to the SGW-service pod.
6	The P-T3 transaction is completed. The SGW-service pod sends the Create Session Request to the S5-SGW-EGRESS-GTP-EP with S11-U and S5-U TEID details.
7	The E-T4 transaction is started. The Create Session Response is validated and the S5-U remote TEID is updated in the PDN. The S5-SGW-EGRESS-GTP-EP sends the Create Session response to the SGW-service pod.
8	The E-T4 transaction is completed. The SGW-service pod sends the Sx Modify Request to the PFCP-EP.
9	The P-T5 transaction is started. The PFCP-EP sends the Sx Modify Response to the SGW-service pod on expiry of the timer T5.
10	The P-T5 and S-T2 transaction is completed. The GTP-EP receives the Create Session Response from the SGW-service pod.
11	The E-T1 transaction is completed. The MME receives the Create Session Response from the S11-GTP-EP.
12	The SGW-service pod updates the created session in CDL endpoint (database) with new PDN information.
13	The MME sends the Modify Bearer Request with TEID to the S11-GTP-EP.
14	The E-T1 transaction is started. The S11-GTP-EP routes the Modify Bearer Request to the SGW-service pod. The S11-GTP-EP sends the Modify Bearer Request to the SGW-service pod.

Step	Description
15	The S-T2 transaction is started. The SGW-service pod sends the Sx Modify Request to the PFCP-EP pod on expiry of the timer S-T2.
16	The P-T3 transaction is started. The PFCP-EP sends the Sx Modify Response to the SGW-service pod on expiry of the timer T3.
17	R-T3 transaction is completed. The SGW-service pod sends the Modify Bearer Response to the MME.
18	S-T2 transaction is completed The SGW-service pod sends the update session to the CDL endpoint.

UE-requested or the MME-requested PDN Disconnection Call Flow

This section describes the UE or the MME-requested PDN disconnection call flow.

Figure 96: UE-requested or the MME-requested PDN Disconnection Call Flow

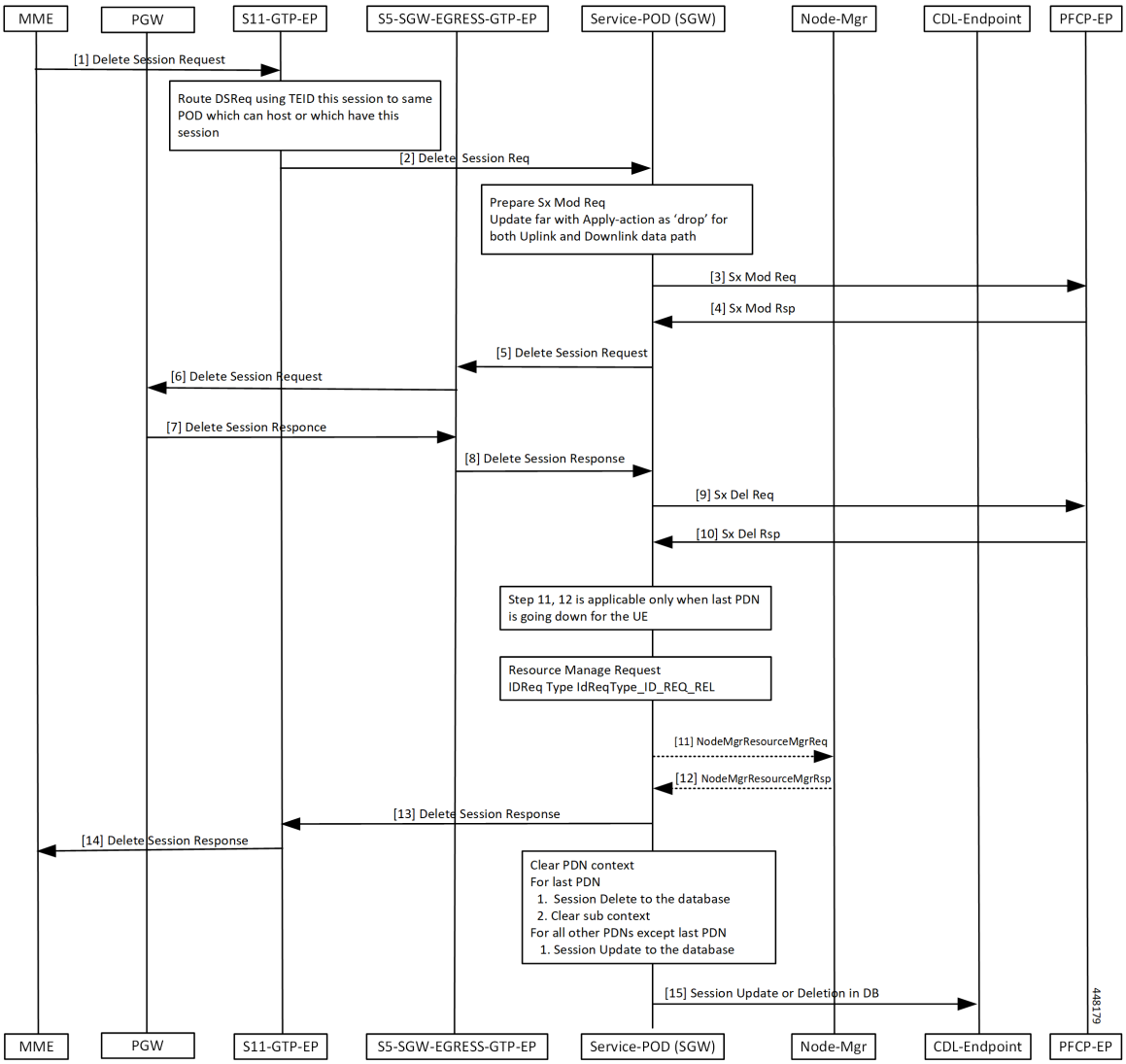


Table 179: UE-requested or the MME-requested PDN Disconnection Call Flow Description

Step	Description
1	The MME sends the Delete Session Request to the S11-GTP-EP for other PDN disconnection.
2	The GTP-EP decodes the received UDP message and converts the message into the gRPC message. Based on the TEID value, the gRPC message is forwarded to the SGW-service pod which can handle the UE session.
3	The SGW-service pod finds the subscriber context information as per the local ingress TEID. The SGW-service pod validates the Delete Session Request content. The SGW-service pod sends the Sx Modify Request to the PFCP-EP with apply action as DROP to drop the uplink or downlink packets at the SGW-U.

Step	Description
4	The PFCP-EP sends the Sx Modify Response to the SGW-service pod.
5	The SGW-service pod forwards the Delete Session Request to the S5-SGW-EGRESS-GTP-EP.
6	The S5-SGW-EGRESS-GTP-EP forwards the Delete Session Request to the PGW through the UDP proxy.
7	The PGW sends the Delete Session Response to the S5-SGW-EGRESS-GTP-EP.
8	The S5-SGW-EGRESS-GTP-EP forwards the Delete Session Response to the SGW-service pod.
9	The SGW-service pod validates the Delete Session Response, and sends the Sx Delete Request to the PFCP-EP.
10	The SGW-service pod receives the Sx Delete Session Response from the PFCP-EP.
11	For the last PDN, the SGW-service pod sends the NodeMgrResourceManager Request for the ID release to the NodeManager.
12	The Node Manager releasees the ID and sends the acknowledgment to the SGW-service pod.
13	The SGW-service pod sends the Delete Session Response to the S11-GTP-EP.
14	The S11-GTP-EP forwards the Delete Session Response to the MME.
15	The SGW-service pod • Updates the session in database for the delete PDN information for other than the last PDN • Sends the delete session message to the database for the last PDN

PGW-requested Disconnection Call Flow

This section describes the PGW-requested disconnection call flow.

Figure 97: PGW-requested Disconnection Call Flow

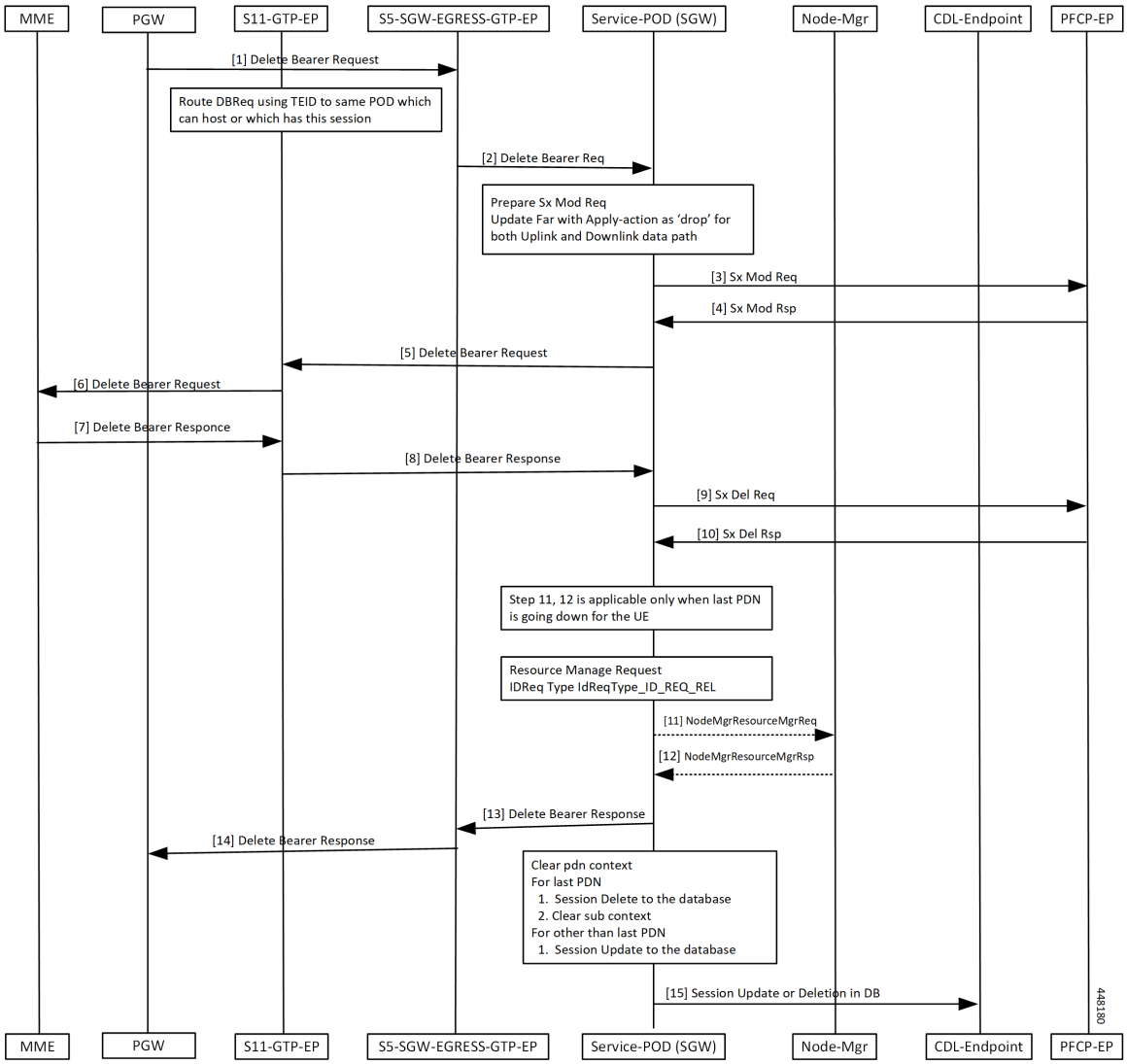


Table 180: PGW-requested Disconnection Call Flow Description

Step	Description
1	The PGW sends the Delete Bearer Request to the S11-GTP-EP for other PDN disconnection.
2	The GTP-EP decodes the received UDP message and converts the message into the gRPC message. Based on the TEID value, the gRPC message is forwarded to the SGW-service pod which can handle the UE session.
3	The SGW-service pod finds the subscriber context information as per the local ingress TEID. The SGW-service pod validates the Delete Bearer Request content. The SGW-service pod sends the Sx Modify Request to the PFCP-EP with apply action as DROP to drop the uplink or downlink packets at the SGW-U.

Step	Description
4	The PFCP-EP sends the Sx Modify Response to the SGW-service pod.
5	The SGW-service pod forwards the Delete Bearer Request to the S5-SGW-EGRESS-GTP-EP.
6	The S5-SGW-EGRESS-GTP-EP forwards the Delete Bearer Request to the PGW through the UDP proxy.
7	The PGW sends the Delete Bearer Response to the S5-SGW-EGRESS-GTP-EP.
8	The S5-SGW-EGRESS-GTP-EP forwards the Delete Bearer Response to the SGW-service pod.
9	The SGW-service pod validates the Delete Bearer Response, and sends the Sx Delete Request to the PFCP-EP.
10	The SGW-service pod receives the Sx Delete Bearer Response from the PFCP-EP.
11	For the last PDN, the SGW-service pod sends the NodeMgrResourceManager Request for the ID release to the NodeManager.
12	The Node Manager releasees the ID and sends the acknowledgment to the SGW-service pod.
13	The SGW-service pod sends the Delete Bearer Response to the S11-GTP-EP.
14	The S11-GTP-EP forwards the Delete Bearer Response to the MME.
15	The SGW-service pod • Updates the Bearer in database for the delete PDN information for other than the last PDN • Sends the delete Bearer message to the database for the last PDN



CHAPTER 40

Performance Optimization Support

- Feature Summary and Revision History, on page 450
- Feature Description, on page 453
- Async BG-IPC from GTPC-EP towards SGW-Service, on page 453
- Batch ID Allocation, Release, and Reconciliation Support, on page 453
- Cache Pod Optimization, on page 456
- CDL Flush Interval and Session Expiration Tuning Configuration, on page 456
- DDN Call Flow Optimization, on page 457
- DDN Timeout Configuration, on page 462
- Domain-based User Authorization Using Ops Center, on page 463
- Edge Echo Implementation, on page 465
- ETCD Peer Optimization Support, on page 467
- Optimized GTPv2 Encoder and Decoder, on page 468
- GTPC Endpoint with GR Split, on page 470
- GTPC Endpoint Interface Split with S11 and S5 , on page 471
- GTPC IPC Cross-rack Support, on page 473
- Interservice Pod Communication, on page 481
- MBR Call Flow Optimization, on page 484
- Maintenance Mode, on page 493
- Partial CDL Update for Idle-Active Call Flow, on page 495
- PFCP Session Report with DLDR Throttling Support, on page 497
- Throttling Support for Create Session Requests on S11 Interface, on page 500
- Resiliency Handling, on page 503
- Roaming Peer Path Management Optimization, on page 506
- Flag DB Database Updates, on page 514
- UDP Proxy Functionality Merged into Protocol Microservices, on page 516

Feature Summary and Revision History

Summary Data

Table 181: Summary Data

Applicable Products or Functional Area	cnSGW-C
Applicable Platforms	SMI

Feature Default Setting	Async BG IPC from GTPC-EP towards SGW-Service: Enabled – Always-on Batch ID Allocation, Release, Reconciliation Support: Disabled – Configuration required to enable Cache Pod Optimization CDL Flush Interval and Session Expiration Tuning Configuration: Enabled – Configuration required to disable DDN Call Flow Optimization: Disabled – Configuration required to enable DDN Timeout Configuration: Disabled – Configuration required to enable Edge Echo Implementation: Enabled – Always-on ETCD Peer Optimization Support: Enabled - Always-on Flag the DB Database Updates: Enabled – Always-on GTPC Interface Split: Disabled – Configuration required to enable GTPC IPC Cross-rack Support: Disabled – Configuration required to enable Interservice Pod Communication: Disabled – Configuration required to enable Maintenance Mode: Disabled – Configuration required to enable MBR Call Flow Optimization: Disabled – Configuration required to enable Optimized GTPv2 Encoder and Decoder: Enabled – Always-on Partial CDL Update for Idle Active Call Flow: Enabled – Always-on PFCP Session Report with DLDR Throttling Support: Disabled – Configuration required to enable Resiliency Handling: Disabled – Configuration required to enable Roaming Peer Path Management Optimization: Disabled – Configuration required to enable UDP Proxy functionality merge into Protocol microservices: Enabled – Configuration required to disable
Related Documentation	Not Applicable

Revision History

Table 182: Revision History

Revision Details	Release
Support for the following sub-features were introduced: • Added support for cache pod optimization. • PFCP Session Report with DLDR Throttling Support • Resiliency Handling	2023.01.0
Support for the following sub-features were introduced: • Batch ID Allocation, Release, and Reconciliation • CDL Flush Interval and Session Expiration Tuning Configuration • DDN Call Flow Optimization • DDN Timeout Configuration • Edge Echo Implementation • ETCD Peer Optimization Support • Flag the DB Database Updates • GR Split • GTPC Interface Split • GTPC IPC Cross-rack Support • Interservice Pod Communication • Introduced support for IPv6. • Maintenance Mode • Optimization of Modify Bearer Request and Response (MBR) call flows • Optimized GTPv2 Encoder and Decoder is provided for additional Request and Response messages. • UDP Proxy and GTPC-EP Merge • UDP Proxy and PFCP-EP Merge	2022.04.0
First introduced.	2021.02.3

Feature Description

This chapter describes about the performance optimization features.

Some of the performance optimization features are common across cnSGW-C and SMF.

For complete information on SMF features, see the *UCC 5G SMF Configuration and Administration Guide*.

Async BG-IPC from GTPC-EP towards SGW-Service

Feature Description

cnSGW-C supports Asynchronous BG-IPC call from GTPC-EP to cnSGW-C, by consuming less resources of the GTPC-EP pod.

For Async BG-IPC call, the GTPC-EP pod uses:

- SendRequestWithCallbackAsResponseWithRequestId API of app-infra to send request messages to the SGW-service.
- GetIPCRequestCallbackResponseWithRequestId API to receive response from the SGW-service.

Batch ID Allocation, Release, and Reconciliation Support

Feature Description

The nodemgr allocates a unique ID to the subscriber that is in the attached state. When the subscriber detaches, the unique ID is released to the nodemgr. If the allocation and deallocation procedures increase, the nodemgr performance is impacted and the sgw-service continues to wait longer to complete these procedures.

The Batch ID Allocation, Release, and Reconciliation Support feature provide a mechanism to reduce the interaction between the sgw-service and nodemgr, which in turn optimizes the nodemgr's performance.

How it Works

This section describes how this feature works.

Batch ID Allocation:

Allocation of batch ID involves the following steps:

- The sgw-service manages the ID store by allocating the free IDs to the subscriber. When the IDs are unavailable in the store, the sgw-service sends the Batch ID Allocation Request to nodemgr.
- In response, nodemgr returns a batch of 128 IDs with the ID Reserve Report Interval. The sgw-service updates the ID store with the IDs received from the nodemgr and starts a timer for the ID Reserve Report Interval.

- If all the IDs are used before the duration configured in the ID Reserve Report Interval, sgw-service sends a Batch of ID Allocation Request to the nodemgr with a notification to reserve all IDs from the previous request.
- If the ID Reserve Report Interval timer expires before the sgw-service allocates all the IDs, sgw-service sends the unused IDs back to nodemgr through the Reserve Report Batch operation.

Batch ID Release:

Releasing of the batch ID involves the following steps:

- The sgw-service manages the IDs that the ID store releases for each nodemgr.
- The sgw-service returns the ID to the ID store whenever an ID is deallocated. If the ID store is full, the sgw-service sends a Batch ID Release Request and the released IDs to the respective nodemgr.
- When sgw-service starts adding IDs to the ID store, the ID release timer starts.
- If the ID release timer expires before the batch IDs are releases or the batch is full, sgw-service sends the released IDs to nodemgr.

Batch ID Reconciliation

Batch ID reconciliation occurs when the service pod and the nodemgr pod restarts.

On service pod restart:

1. When the service pod receives the batch IDs and becomes unresponsive before allocating the IDs, the nodemgr does not get the Batch ID Reserve Request causing the ID reserve procedure to time out. In such a scenario, the nodemgr reconciles the unreserved or unallocated IDs with CDL. The IDs that are not allocated to the subscribers are released to the ID store.
2. The service pod collects the IDs that are released and if it becomes unresponsive before releasing them to the nodemgr. In this scenario, the IDs are dropped.

On nodemgr pod restart:

1. The IDs existing in the in-flight Batch ID Reserve Request and Batch ID Release Request are dropped.
2. The nodemgr notifies cachemgr about the allocated IDs in a batch. If nodemgr becomes unresponsive before notifying the IDs to cachemgr, after a restart, nodemgr starts allocating the new IDs. The nodemgr allocated the IDs based on the last allocated ID and the batch size.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   sgw sgw_name
      resmgr-batch-operation [ disable | enable ]
      end
```

NOTES:

resmgr-batch-operation [disable | enable]—Configures the batch operation. By default, **resmgr-batch-operation** is disabled.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics

The following statistics are supported for the Batch ID Allocation and Release Support feature:

- sgw_resource_mgmt_stats—Captures the total number of the cnSGW-C resource management statistics.

  Sample queries:

```
sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",
id_req_type="id_batch_alloc",instance_id="0",service_name="sgw-service",status="attempted"}
 3
:sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",
id_req_type="id_batch_alloc",instance_id="0",service_name="sgw-service",status="success"}
 3

sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1"
,id_req_type="id_batch_dealloc",instance_id="0",service_name="sgw-service",status="attempted"}
 2
sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",id_req_type=
"id_batch_dealloc",instance_id="0",service_name="sgw-service",status="success"} 2

sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",
id_req_type="id_batch_dealloc_timeout",instance_id="0",service_name="sgw-service",status="attempted"}
 1
:sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",id_req_type
="id_batch_dealloc_timeout",instance_id="0",service_name="sgw-service",status="success"}
 1

sgw_resource_mgmt_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",
id_req_type="id_batch_release_timeout",instance_id="0",service_name="sgw-service",status="attempted"}
 1
-:sgw_resource_mgmt_stats{app_name="smf",cluster="Local",d-ata_center="DC",gr_instance_id="1",id_req_type
="id_batch_release_timeout",instance_id="0",service_name="sgw-service",status="success"}
 1
```

- nodemgr_rmgr_batch_reconcile_stats—Captures the total count of batches that are sent for reconciliation.

  Sample queries:

```
nodemgr_rmgr_batch_reconcile_stats{app_name="smf",cluster="Local",data_center="DC",instance_id="0",
service_name="nodemgr",status="success"} 1
```

- nodemgr_resource_mgmt_resp_stats—Captures the total number of IDs released due to reconciliation.

  Sample queries:

```
nodemgr_resource_mgmt_resp_stats{app_name="smf",cluster="Local",data_center="DC",error="",
gr_instance_id="0",instance_id="0",ip_ver_type="IP_TYPE_NONE",req_type="ID_REQ_REL_RECONCILE",
service_name="nodemgr",status="success"} 16
```

For more information on bulk statistics support, see *UCC Serving Gateway Control Plane Function Metrics Reference*.

Cache Pod Optimization

Feature Description

The cnSGW-C supports the cache pod optimization to reduce the cache pod query at the GTPC endpoint.

The get affinity query is used to receive the affinity information in an outgoing request or response message toward the GTPC endpoint. With this optimization, the GTPC endpoint pod doesn't send the query to the cache pod for the upcoming request messages.

To receive this affinity information, an affinity query is used in an outgoing request or response message toward the GTPC endpoint. With this optimization, the GTPC endpoint pod doesn't send the query to the cache pod for the upcoming request messages.

In the previous releases, after the cnSGW-C sent out the DDN and received the MBR from the MME, the GTPC endpoint had to send the query to the cache pod to get affinity information. Later, the cnSGW-C used the affinity information so that an MBR can be forwarded to the correct service pod.

With this optimization, you can prevent the extra cache pod query.

CDL Flush Interval and Session Expiration Tuning Configuration

Feature Description

You can modify the default service-pod parameters to fine-tune the throughput performance and optimize the load performance.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   profile sgw sgw_name
      timers [ session-expiration-in-secs session_expiration |
affinity-expiration-in-secs affinity_expiration | session-dbsync-interval-in-ms
 database_sync ]
      end
```

NOTES:

- **session-expiration-in-secs** *session_expiration* —Specify the duration for which the session is cached on service pod. *session_expiration* accepts value in the range of 1-600 milliseconds. The default value is 30 milliseconds.
- **affinity-expiration-in-secs** *affinity_expiration* —Specify the duration for which the session affinity keys are valid on the service pod and other pods. *affinity_expiration* accepts value in the range of 1-1200 seconds. The default value is 80 seconds.

- **session-dbsync-interval-in-ms** *database_sync* —Specify the duration after which the session is synchronized in the database. *database_sync* accepts value in the range of 1-10000 milliseconds. The default value is 500 milliseconds.

Configuration Example

The following is an example configuration.

```
config
     profile sgw sgw1 [ timers session-expiration-in-secs 30 | affinity-expiration-in-secs
 80 | timers session-dbsync-interval-in-ms 500 ]
     end
```

DDN Call Flow Optimization

Feature Description

The Downlink Data Notification (DDN) Call Flow Optimization feature lets you suspend the invoking of the DDN procedure for a specified period. With this feature, the network-initiated service request procedure is scheduled before the DDN is received and the UE is moved to the active state. In case of the timer expiry, if the UE ID is active, or the Modify Bearer Request is in progress, the cnSGW ignores the DDN procedure.

How it Works

This section describes how this feature works.

cnSGW-C invokes the DDN procedure in the following scenarios:

- A session rapidly moves to the idle state causing the Uplink and Downlink data to trigger simultaneously.
- When a subscriber is in the idle state and the Downlink data is received, cnSGW-C sends a DDN to MME. The DDN notifies MME to page the UE and change the UE state to active.
- When a subscriber is in the idle state and the Uplink data is received, the MME sends a Modify Bearer Request to SGW to change the UE state to active.
- If the SGW service has initiated DNN and it receives the Modify Bearer Request, the service aborts the DDN procedure and processes the Modify Bearer Request to change the UE state to active.

Call Flows

This section describes the key call flows for this feature.

Current Downlink Data Notification Handling Call Flow

This section describes the Current Downlink Data Notification Handling call flow.

Figure 98: Current Downlink Data Notification Handling Call Flow

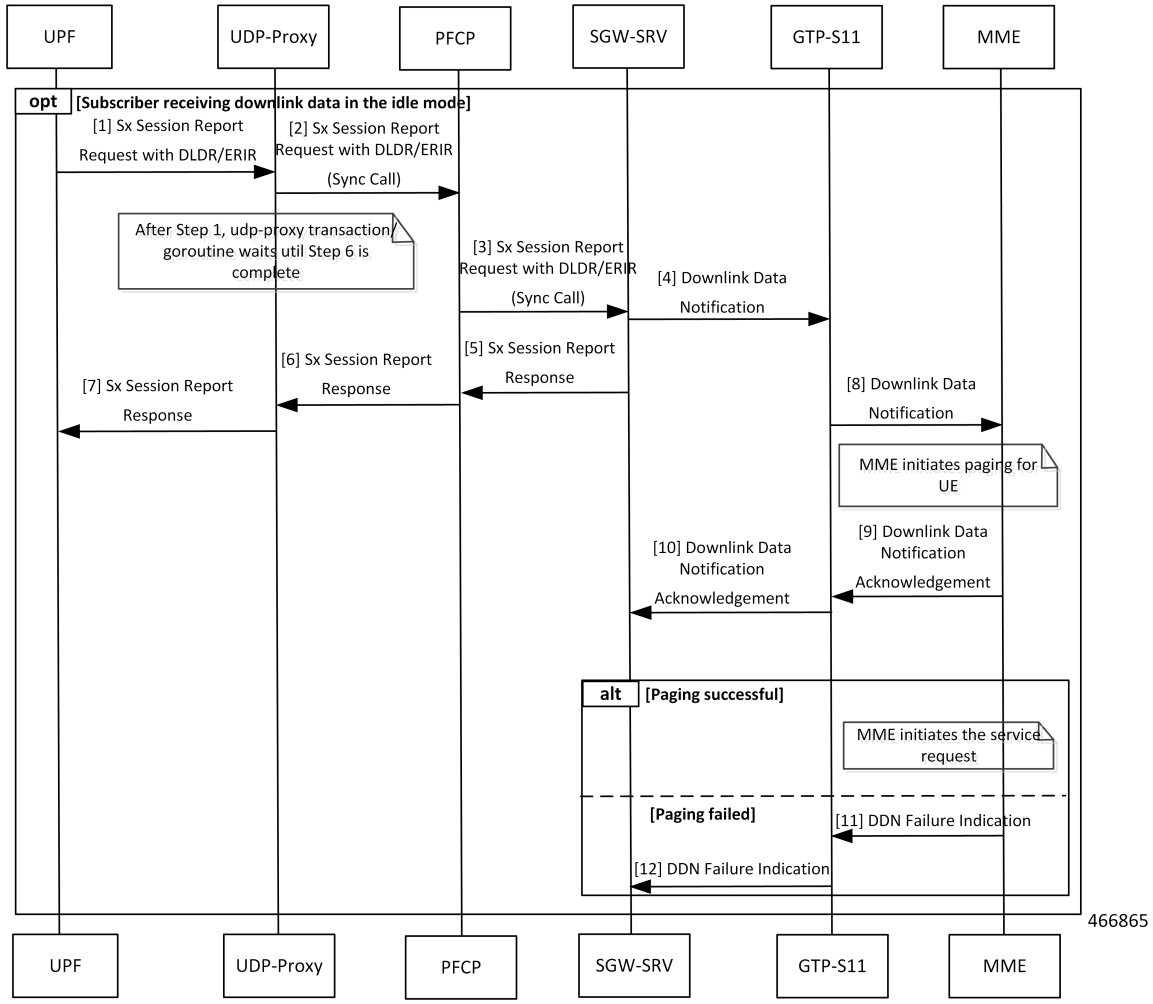


Table 183: Current Downlink Data Notification Handling Call Flow Description

Step	Description
1	UPF sends a Sx Session Report Request with DLDR or ERIR to UDP-Proxy.
2	UDP-Proxy sends the Sx Session Report Request with DLDR or ERIR to PFCP.
3	PFCP sends the Sx Session Report Request with DLDR or ERIR to SGW-SRV.
4	The SGW-SRV sends the Downlink Data Notification (DDN) to GTP-S11.
5	SGW-SRV sends Sx Session Report Response to PFCP.
6	PFCP sends the Sx Session Report Response to UDP-Proxy.
7	UDP-Proxy sends the Sx Session Report Response to UPF.
8	GTP-S11 sends the DDN Notification to MME.

Step	Description
9	MME initiates a paging for UE and sends the DDN Acknowledgment to GTP-S11.
10	GTP-S11 sends the DDN Acknowledgment to SGW-SRV.
11	When the paging fails, MME sends the DDN Failure Indication to GTP-S11.
12	GTP-S11 sends the DDN Failure Indication to SGW-SRV.

DDN Handling with Internal DDN Delay Timer Call Flow

This section describes the DDN Handling with Internal DDN Delay Timer call flow.

Figure 99: DDN Handling with Internal DDN Delay Timer Call Flow

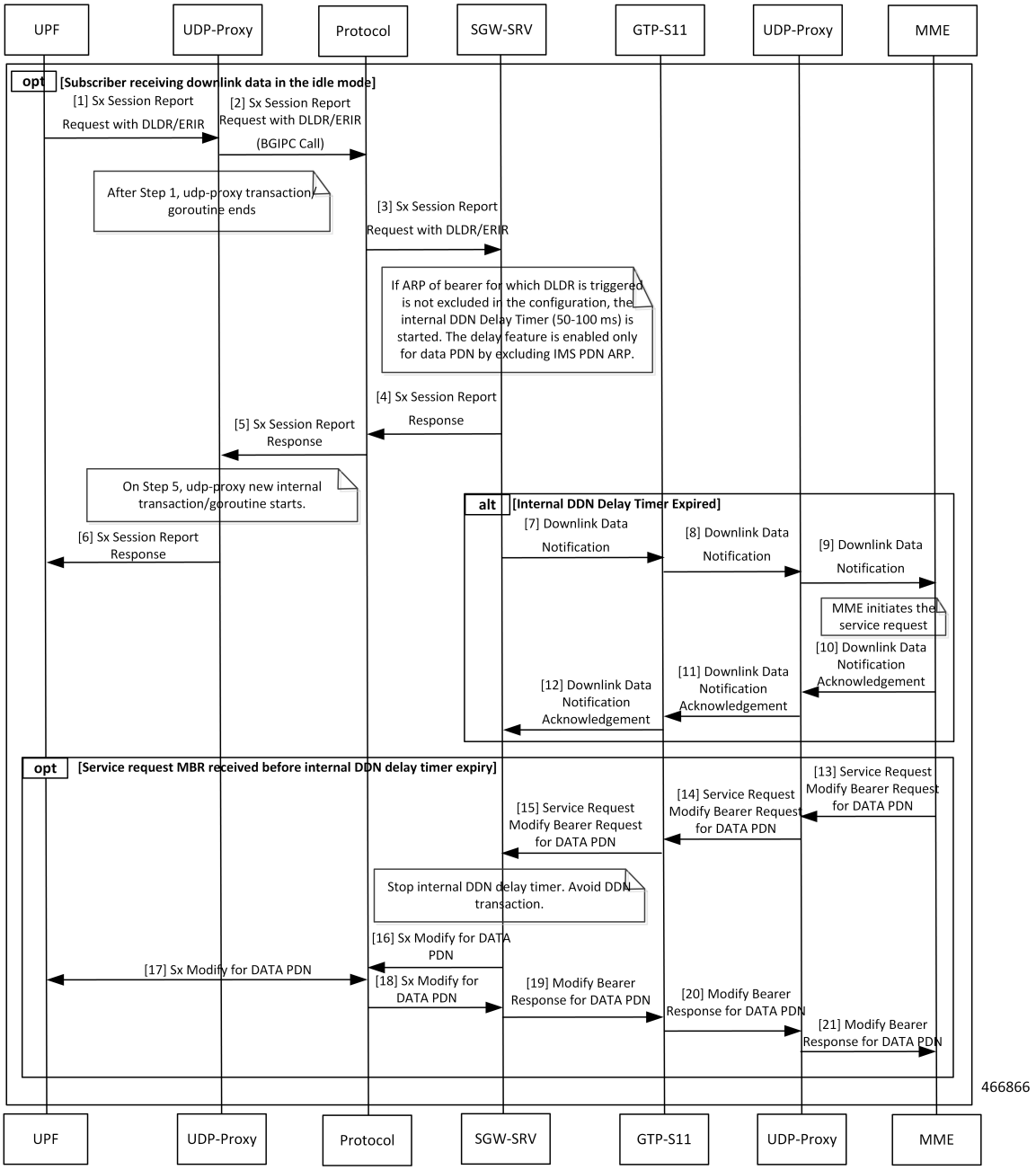


Table 184: DDN Handling with Internal DDN Delay Timer Call Flow Description

Step	Description
1	UPF sends a Sx Session Report Request with DLDR/ERIR to UDP-Proxy.
2	UDP-Proxy sends the Sx Session Report Request with DLDR/ERIR (BGIPC Call) to Protocol.
3	Protocol sends the Sx Session Report Request with DLDR/ERIR to SGW-SRV.

Step	Description
4	SGW-SRV sends a Sx Session Report Response to Protocol.
5	Protocol sends the Sx Session Report Request to UDP-Proxy.
6	UDP-Proxy sends the Sx Session Report Request to UPF.
7	SGW-SRV sends the Downlink Data Notification to GTP-S11.
8	GTP-S11 sends the Downlink Data Notification to UDP-Proxy.
9	UDP-Proxy sends the Downlink Data Notification to MME.
10	After MME initaites paging for UE, MME sends the Downlink Data Notification Acknowledgement to UDP-Proxy.
11	UDP-Proxy sends the Downlink Data Notification Acknowledgement to GTP-S11.
12	GTP-S11 sends the Downlink Data Notification Acknowledgement to SGW-SRV.
13	MME sends the Modify Bearer Request for DATA PDN to UDP-Proxy.
14	UDP-Proxy sends the Modify Bearer Request for DATA PDN to GTP-S11.
15	GTP-S11 sends the Modify Bearer Request for DATA PDN to SGW-SRV.
16	SGW-SRV sends the Modify Bearer Request for DATA PDN to Protocol.
17	Protocol and UPF processes the Sx Modify Bearer Request for DATA PDN.
18	Protocol sends Sx Modify Response for DATA PDN to SGW-SRV.
19	SGW-SRV sends Sx Modify Response for DATA PDN to GTP-S11.
20	GTP-S11 sends Sx Modify Response for DATA PDN to UDP-Proxy.
21	UDP-Proxy sends Sx Modify Response for DATA PDN to MME.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   profile sgw sgw_name
      ddn { delay-exclude-arplist number_priorities | delay-timer delay_duration
 }
       end
```

NOTES:

- **delay-exclude-arplist** *number_priorities*—Specify the priority-level for allocation and retention priorities [1-15] that must be excluded from delaying the DDN. *number_priorities* can accept a maximum of eight entries.

- **delay-timer** *delay_duration*—Specify the duration for which the DDN procedure is delayed. *delay_duration* accepts duration in milliseconds 0–5000. The default duration is 0 which indicates that the timer is disabled.

Configuration Example

The following is an example configuration.

```
config
    profile sgw sgw1
       ddn delay-timer 100 delay-exclude-arplist [ 3 4 ]
       end
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics

The following statistics are supported for the DDN Call Flow Optimization feature:

sgw_tmr_stats—The internal DDN delay timer for stop, start, and expired states.

Query:

```
sgw_tmr_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",service_name="sgw-service",status="expired",timer_type="internal_ddn_delay"}
1

sgw_tmr_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",instance_id="0",
service_name="sgw-service",status="start",timer_type="internal_ddn_delay"} 2

sgw_tmr_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",instance_id="0",
service_name="sgw-service",status="stop",timer_type="internal_ddn_delay"} 1
```

DDN Timeout Configuration

Feature Description

cnSGW-C lets you configure the DDN Timeout and Peer Not Responding configuration through the cnSGW-C Ops Center.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   profile sgw sgw_name
      ddn timeout-purge-session { true | false }
      end
```

NOTES:

ddn timeout-purge-session { true | false }—Configures the session when the MME does not send the DDN acknowledgment. The default value is false.

Domain-based User Authorization Using Ops Center

Feature Description

5GSA GW and cnSGW-C support domain-based user authorization using the Ops Center. To control the access on a per-user basis, use the TACACS protocol in Ops Center AAA. This protocol provides centralized validation of users who attempt to gain access to a router or NAS.

Configure the NETCONF Access Control (NACM) rules in the rule list. Then, map these rules in the Ops center configuration to map the group to appropriate operational authorization. Use the configurations that are based on the following criteria and products:

- With the NACM rules and 5GSA GW domain-based group, configure the Ops center to allow only access or update 5GSA GW-based configuration.
- With the NACM rules and cSGW-C domain-based group, configure the Ops center to allow only access or update cSGW-C-based configuration.
- With the NACM rules and cSGW-C domain-based group, configure the Ops center to allow only access or update CCG-based configuration.

Note The NSO service account can access the entire configuration.

How it Works

To support this feature configuration in Ops Center, the domain-based-services configuration is added in the TACACS security configuration. The TACACS flow change works in the following way:

- If you have configured the **domain-based-services** parameter, then the configured user name that is sent to the TACACS process, splits user ID into user ID and domain. The split character, which is a domain delimiter, is configured in domain-based-services. These split characters can be "@", "/", or "\" and are used in the following format to get the domain and user ID information.
  - @ — <user id>@<domain>
  - / — <domain>/<user id>
  - \ — <domain>\<user id>
- The TACACS authenticates and authorizes as per the existing flow. However, if the domain-based-services feature is enabled and TACACS authenticates and authorizes the user, following steps are added to the TACACS flow procedure.
  - If Network Services Orchestrator (NSO) logs in as the NSO service account, then that session receives a specific NACM group that you configured in **domain-based-services nso-service-account group** *group-name*. This functionally is the same as the way NSO works.

- If the specified domain exists in the group mapping, then the NACM group that you configured in **domain-based-services domain-service** *domain* **group** *group-name* is applied.
- If the user does not have a domain or the domain does not exist in the domain to group mapping, then **no-domain** NACM group that you configured in **domain-based-services no-domain group** *group-name* is applied. If the **no-domain** configuration does not exist, then the user value is rejected.

To enable this feature, you must configure the **domain-based-services** CLI command with the following options:

- NSO service account
- Domain service
- Domain delimiter
- No domain

Feature Configuration

To enable domain-based user authorization using Ops Center, use the following sample configuration:

```
config
  tacacs-security domain-based-services [ domain-delimiter delimiter_option
 | domain-service domain_service_name [ group service_group_name ] | no-domain
group service_group_name | nso-service-account [ group service_group_name | id
service_account_id ] ]
  end
```

NOTES:

- **domain-based-services [domain-delimiter** *delimiter_option* **| domain-service** *domain_service_name* **[group** *service_group_name* **] | no-domain group** *service_group_name* **| nso-service-account [group** *service_group_name* **| id** *service_account_id* **]]**: Configure the required domain-based-services value. The **domain-based-services** includes the following options:
  - **domain-delimiter**: Specify the delimiter to use to determine domain. This option is mandatory and allows the following values:
    - @—If domain-delimiter is "@", the user value is in the format: <user>@<domain>.
    - /—If domain-delimiter is "/", the user value is in the format: <domain>/<user>.
    - \—If domain-delimiter is "\", the user value is in the format: <domain>\<user>.
  - **domain-service**: Specify the list of domains and their group mapping. The key is the name of the domain and group is the group that is assigned to the domain. You must configure at least one option in this list.
  - **no-domain**: Specify the group that has no domain or if the domain is unavailable in the domain-service mapping, then this group is sent in the accept response.
  - **nso-service-account**: Specify the NSO service account that has the ID and group. If you configure this parameter, then you must configure the ID and group fields. The ID and group must have string values.

Configuration Example

The following is an example of the domain-based user authorization in the tacacs-security mode:

```
config
   tacacs-security domain-based-services nso-service-account id nsid
      tacacs-security domain-based-services nso-service-account group nso-group
   tacacs-security domain-based-services no-domain group read-operational
   tacacs-security domain-based-services domain-delimiter @
   tacacs-security domain-based-services domain-service etcd
      group etcd
exit
tacacs-security domain-based-services domain-service sgw
   group sgw_1
exit
tacacs-security domain-based-services domain-service smf
   group smf
exit
```

Configuration Verification

To verify the configuration, use the following show command:

show running-config tacacs-security

The output of this show command displays all the configurations of the domain-based services within the TACACS security.

```
[smf] smf# show running-config tacacs-security
tacacs-security service smf
tacacs-security server 1
address 209.165.200.234
key $8$+twbdL2ZCgmjVswgp7kFJp8+SMXDjQRTZgoPVa3oEwY=
exit
tacacs-security domain-based-services nso-service-account id nsid
tacacs-security domain-based-services nso-service-account group nso-group
tacacs-security domain-based-services no-domain group read-operational
tacacs-security domain-based-services domain-delimiter @
tacacs-security domain-based-services domain-service etcd
group etcd
exit
tacacs-security domain-based-services domain-service sgw
group sgw_1
exit
tacacs-security domain-based-services domain-service smf
group smf
exit
```

Edge Echo Implementation

Feature Description

In a nonmerged mode, the udp-proxy pod acts as an endpoint, and the gtpc-ep responds to the Echo Requests from the peer node.

The gtpc-ep experiences traffic when the system receives a high number of inputs CEPS leading to a discrepancy between the rate at which gtpc-ep picks up the messages from udp-proxy and the rate at which udp-proxy gets the messages.

If the gtpc-ep is loaded, the queue between the udp-proxy and gtpc-ep gets full, and some of the messages at udp-proxy might get dropped. The peer detects path failure if these are Echo Request messages because an Echo Response is not received. Further, the peer clears all the sessions sent to the sgw-service.

How it Works

This section describes how this feature works.

Nodemgr processes the Echo Request in the following steps:

- The nodemgr preserves a self-restart counter cache for each GR instance ID and the GTPC peer.
- When the udp-proxy pod receives an Echo Request from a peer and the self-restart counter value is not available in the self-restart counter cache, the udp-proxy pod forwards the Echo Request to gtpc-ep.
- The gtpc-ep sends the self-restart counter as part of the UDP proxy message metadata in the Echo Response. The udp-proxy stores the self-restart counter in the self-restart counter cache. When the udp-proxy receives an Echo Request from a peer, and a self-restart counter value is available in the self-restart counter cache, the udp-proxy sends an Echo Response with the restart counter.
- The udp-proxy forwards the Echo Request message to the gtpc-ep. The gtpc-ep processes the Echo Request and forwards it to nodemgr, if necessary.
- If the peer restart counter value is modified, the nodemgr detects a path failure.
- In the Echo Response, the gtpc-ep sends the self-restart counter in the UDP Proxy Message metadata to the udp-proxy. If the self-restart counter differs from the counter that is stored in the self-restart counter cache, the udp-proxy updates the self-restart counter in the cache and drops the Echo Response received from the gtpc-ep.



Note The Edge Echo feature is not supported when the gtpc-ep is started in the merged mode.

Heartbeat

To handle the Echo Request and Echo Response messages for the GTPV2 interface, a heartbeat queue is implemented between the gtpc-ep and the udp-proxy pod. The heartbeat queue is responsible for handling the HeartBeat Request and HeartBeat Response Messages between the protocol and udp-proxy pod for the PFCP interface.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the Edge Echo Implementation feature:

- Heartbeat queue status:

```
sum(irate(ipc_response_total{rpc_name~=".ipc_stream_hb."}[10s])) by
(service_name,
instance_id, status, status_code, rpc_name, dest_host)
```

- Check the EdgeEcho messages:

```
sum(irate(udp_proxy_msg_total{ message_name ="edge_echo"}[30s])) by
(message_name,
message_direction, status)
```

To enable the Heartbeat queue and EdgeEcho messages statistics, configure the trace-level statistics for udp_proxy_msg_total using the following:

```
infra metrics verbose application
   metrics udp_proxy_msg_total level trace
   exit
```

Note Enabling the heartbeat and EdgeEcho messages statistics may lead to a performance degradation on the udp-proxy pod.

ETCD Peer Optimization Support

Feature Description

When large numbers of GTPC peers are connected with 5GSA GW or cnSGW-C, the performance of ETCD is impacted. Each peer is a considered as a record in the ETCD, and the timestamp is updated every 30 seconds for each peer. This causes continuous updates on ETCD and generates huge traffic that impacts the overall system performance.

The ETCD Peer Optimization feature facilitates optimization in peer management and enables reduced performance impact on ETCD.

How it Works

This section describes how this feature works.

Instead of considering each peer as an ETCD record entry, several peers are grouped as a peer group based on the hash value of the IP address of each peer. This reduces the number of entries in ETCD. By default, a maximum of 200 peer groups can be created. For any changes related to a peer in a peer group:

- For a new peer, the peer group is persisted immediately in ETCD.
- For the change in timestamp for existing peers, the peer group is updated once every 3 seconds. This update:

- Results in a cumulative group update for many peers that have undergone timestamp change within each peer group.
- Reduces frequent updates to ETCD.

Optimized GTPv2 Encoder and Decoder

Feature Description

cnSGW-C provides an optimized GTPv2 encoder and decoder for:

- Modify Bearer Request and Response messages when the subscriber is moving to ACTIVE state after receiving the Modify Bearer Request message.
- Release Access Bearer Request and Response messages when the subscriber is moving to IDLE state on receiving the Release Access Bearer Request message.

The optimized GTPv2 encoder and decoder is provided for the following messages:

- Bearer Resource Command
- Change Notification Request and Response
- Create Bearer Request and Response
- Create IDFT Request and Response
- Create Session Request and Response
- Delete Bearer Command
- Delete Bearer Failure Indication
- Delete Bearer Request and Response
- Delete IDFT Request and Response
- Delete Session Request and Response
- Downlink Data Notification Acknowledgment
- Downlink Data Notification Failure Indication
- Download Datalink Notification Request
- Echo Request and Response
- Modify Bearer Request and Response
- Modify Bearer Command
- Modify Bearer Failure Indication
- Update Bearer Request and Response

Feature Configuration

To configure this feature on S11, S5, and S5e interfaces, use the following configuration:

```
config
   instance instance-id instance_id
      endpoint gtp
         replicas replica_count
         vip-ip ipv4_address  vip-port ipv4_port_number
         vip-ipv6 ipv6_address  vip-ipv6-port ipv6_port_number
         dual-stack-transport { true | false }
         enable-go-encdec { true | false }
         interface interface_name
            enable-go-encdec { true | false }
            end
```

NOTES:

- **enable-go-encdec { true | false }**—Enable the Go language-based GTPv2 encoder and decoder for the interface.
- **dual-stack-transport { true | false }**—Enable the dual stack feature that allows you to specify IPv6 or IPv4 address. Specify true to enable this feature.

Configuration Example

The following is an example configuration.

```
config
    instance instance-id 1
       endpoint gtp
          replica 2
          vip-ip 209.165.200.224 vip-port 2022
          vip-ipv6 ipv6_address 2001:db8:1::2 22
          dual-stack-transport true
          enable-go-encdec false
          exit
       interface s5e
          replica 3
          vip-ip 209.165.200.225 vip-port 2022
          vip-ipv6 ipv6_address 2001:db8:1::3 22
          dual-stack-transport true
          enable-go-encdec true
          exit
       interface s11
          replica 3
          vip-ip 209.165.200.226 vip-port 2022
          vip-ipv6 ipv6_address 2001:db8:1::4 22
          dual-stack-transport true
          enable-go-encdec true
          exit
       interface s5
          replica 3
          vip-ip 209.165.200.227 vip-port 2022
          vip-ipv6 ipv6_address 2001:db8:1::5 22
          dual-stack-transport true
          enable-go-encdec true
          end
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the Optimized GTPv2 Encoder and Decoder feature:

Grafana Query for Go based encoder decoder:

Query:

```
sum(irate(gtpc_golang_enc_dec_stats{namespace="$namespace"}[60s]))
by (gtpc_msg_type, gtpc_msg_operation, gtpc_msg_status)
```

Legend:

```
{{gtpc_msg_type}}-{{gtpc_msg_operation}}-{{gtpc_msg_status}}
```

GTPC Endpoint with GR Split

Feature Description

The GR Split feature enables handling the scaled GTP traffic and facilitates the optimal use of the CPU. The GR Split feature starts multiple active instances of GTPC-EP and performs traffic split which is based on GR instances. This helps in aiding the UDP proxy bypass feature.

How it Works

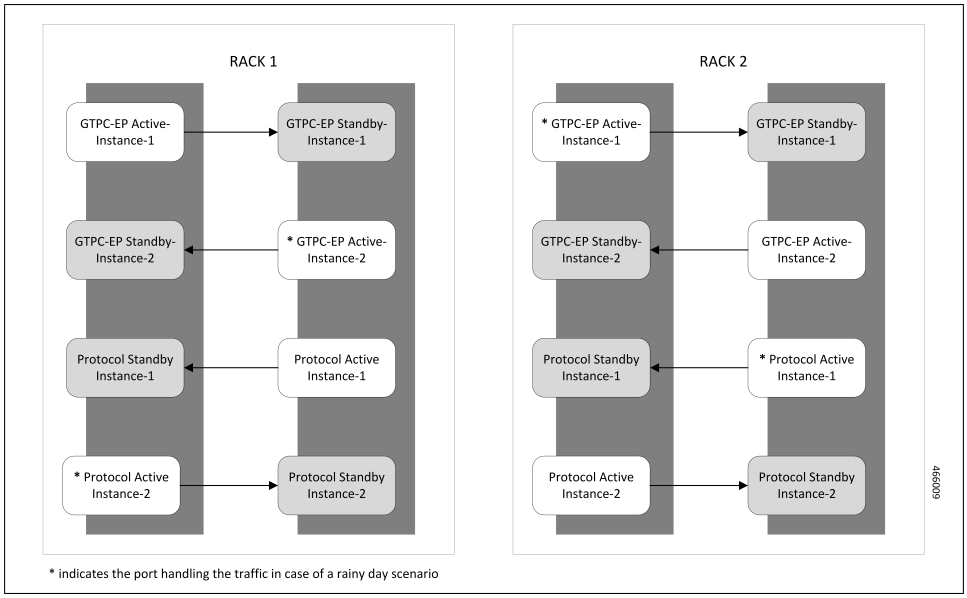
This section describes how this feature works.

The UDP Proxy merge mode is a prerequisite for this feature.

- For a sunny-day scenario, RACK1 (GTPC-EP Active Instance-1) and RACK2 (GTPC-EP Active Instance-2) handle the traffic from GR-Instance-1 and GR Instance-2, respectively.
- For a rainy-day scenario, the GR Instance-1 (S11, S5E, S5) and GR Instance-2 (S11, S5E, S5) traffic splits between two GTPC-EP instances.

  In the rainy-day scenario, assuming RACK2 is down, RACK1 handles all the traffic for GR Instance-1 and GR Instance-2. With the GR Split implementation, the GTPC-EP Active Instance-1 handles GTP traffic for all the interfaces for GR Instance-1, and GTPC-EP Active Instance-2 handles all the GTP traffic for GR Instance-2.

Figure 100: GTPC-EP with Merged Mode and GR Split



GTPC Endpoint Interface Split with S11 and S5

Feature Description

The GTPC Interface Split feature enables splitting the GTPC endpoint based on the interface. cnSGW-C splits the GTPC pod into two interfaces, S11 and S5 which enables handling all GTPC incoming and outgoing traffic for S11 (SGW-Ingress), S5E (SGW-Egress), and S5 (SMF-Ingress).

How it Works

This section describes how this feature works.

The GR Instance Split or UDP Proxy merge mode is a prerequisite for this feature.

This feature is disabled by default. While configuring this feature, provide separate internal and external VIP addresses for S11 and S5 GTPC endpoints, and deploy two GTPC-EP pods.

On configuring the feature, cnSGW-C sends the IPC call from SMF-service, SGW-service, and the Node Manager to the GTPC pod based on the GTPC interface and the GR instance ID.

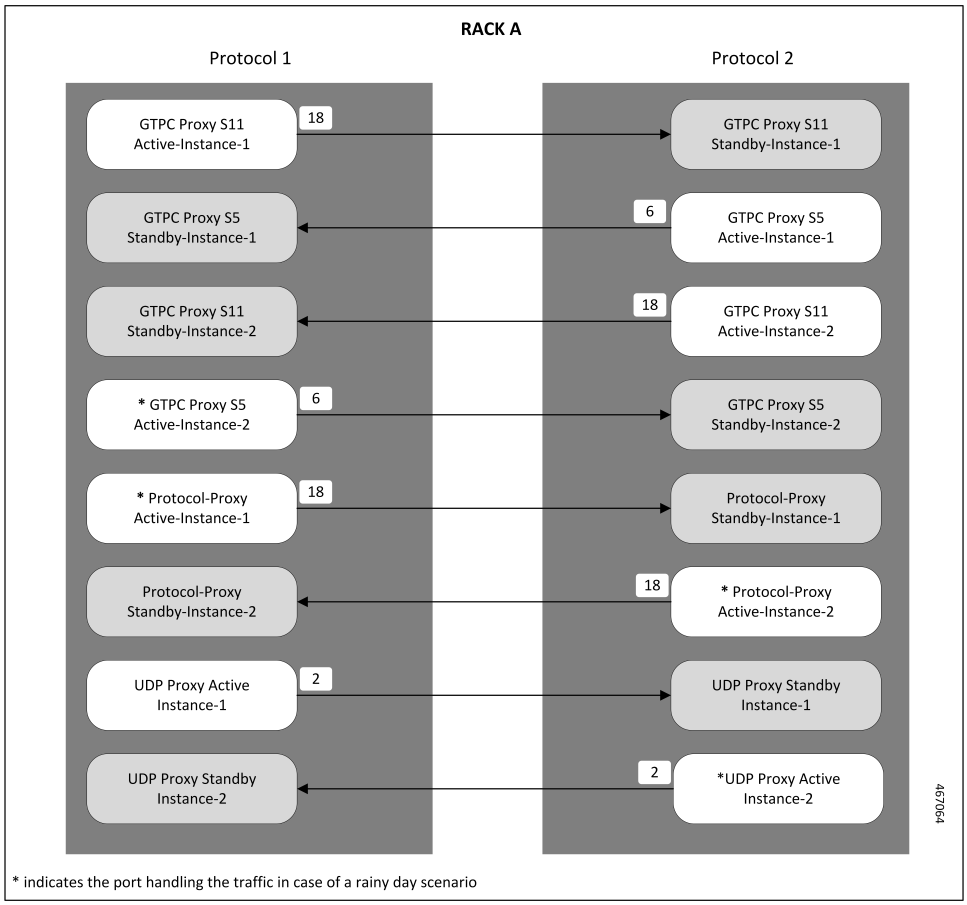
The following are the recommendations for the GTPC interface split feature:

- CPU Core:
  - S5 Interface: 6
  - S11 Interface: 18
  - UDP Proxy: 2

- VIP Configuration: Separate internal and external VIP for S5 and S11 interfaces
- The Dispatcher configuration must be the same configuration as existing before upgrade.

The following diagram indicates the GTPC Interface Split pod layout.

Figure 101: GTPC Interface Split pod layout



Feature Configuration

To configure this feature, use the following configuration:

```
config
  instance instance-id instance_id
    endpoint gtp
    interface s11
      standalone true
      cpu max-process cpu_core_value
      internal-vip internal_vip_address
      vip-ip ipv4_address vip-port ipv4_port_number
      vip-ipv6 ipv6_address vip-ipv6-port ipv6_port_number
      dual-stack-transport { true | false }
      vip-interface vip_interface_value
```

```
      exit
   exit
   interface s5
      vip-ip ipv4_address vip-port ipv4_port_number
      vip-ipv6 ipv6_address vip-ipv6-port ipv6_port_number
      dual-stack-transport { true | false }
      vip-interface vip_interface_value
      end
```

NOTES:

- **standalone true**: Configures the interface to run in standalone mode with a separate pod for the interface.
- **cpu max-process** *cpu_core_value*: Specify the CPU core value for the CPU for the interface. This sets the GO_MAX_PROCS parameter value for the pod.
- **dual-stack-transport { true | false }**—Enable the dual stack feature that allows you to specify IPv6 or IPv4 address. Specify true to enable this feature.

Configuration Example

The following is an example configuration.

```
config
   instance instance-id 1
      endpoint gtp
      interface s11
         standalone true
         cpu max-process 18
         internal-vip 209.165.200.225
         vip-ip 209.165.200.226 vip-interface bd1.gtp.2131
         exit
      exit
      interface s5
      vip-ip 209.165.200.228 vip-interface bd1.gtp.2131
      end
```

GTPC IPC Cross-rack Support

Feature Description

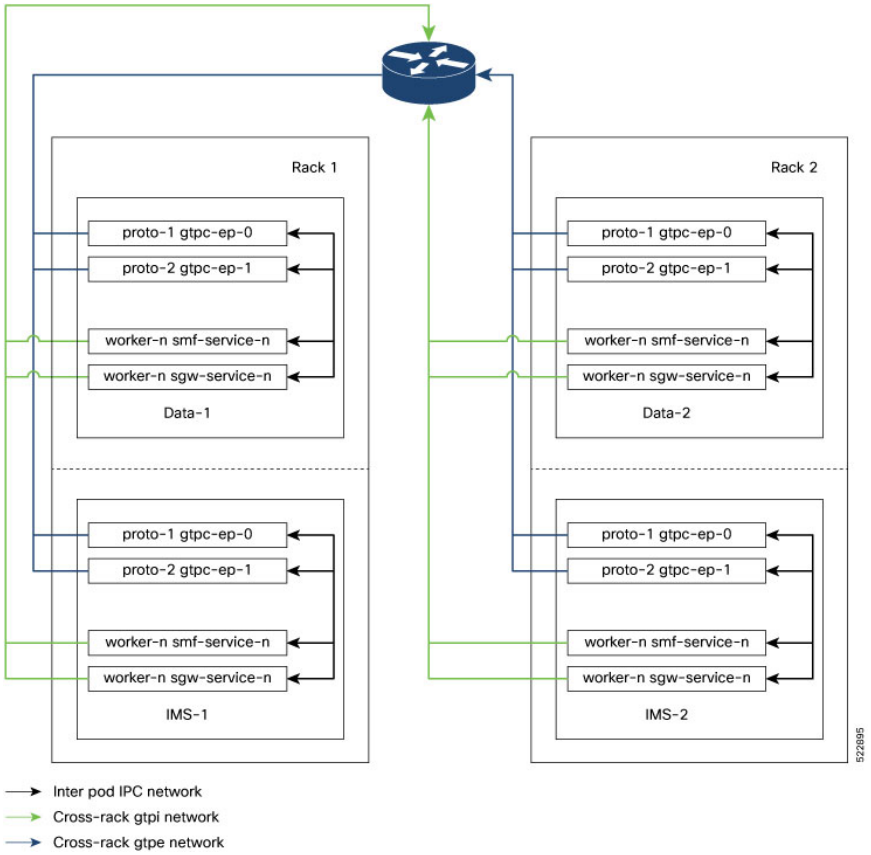
When you perform GR-setup activities with cnSGW-C and SMF, the GTPC message handling can be optimized between these two racks, as in the following scenarios:

- The set of IPC messages from cnSGW-C to SMF service pods flow over `gtpc-ep` pods twice leading to message encoding and decoding overheads.
- Within a GR pair, these IPC messages can avoid one more processing step, if service pods such as cnSGW-C and SMF can route messages to the corresponding peer GTPC nodes directly.

Note Before applying the configuration for enabling GTPC IPC on cnSGW or SMF interfaces, it is required to apply inter-rack routing networks using cluster sync. More configuration is required to add BGP routes for supporting new routable networks across rack servers.

The following figure represents a design of the new network layout that is required for supporting the feature, the core setup activities, and their interconnections.

Figure 102: SGW-GTPC Inter-rack IPC



The following steps are performed for the GTPC message handling optimization between two racks and deploying the cross-rack endpoints:

- The cnSGW-C in IMS-1 Rack-1 routes the IPC request internally to PGW GTPC-EP in DATA-2 Rack-2 passing through the cross-rack GTPI network to the router.
- The router will then use the GTPE network as the next-hop for forwarding requests to the `gtpc-ep` pod.
- The GTPI and GTPE network are new networks added to the Racks during the process of deployment.
- Also, the feature requires internal GTPC IPC messages, which are received on the active `gtpc-ep` pod.
- In this process, the IPC messages from cnSGW-C to SMF service pods flow over the GTPC-EP pods, leading twice to message encoding and decoding outlays.

- Within a GR pair, such IPC messages can avoid one extra hop of processing, if these service pods (cnSGW-C and SMF) can route messages to the corresponding peer GTPC nodes directly.

Note The configured protocol nodes must be in the same VIP group as S5 and S5e VIP groups are deployed.

How it Works

This section describes how this feature works:

- In `SMF-Ops-Center`, you can configure `GTPC-EP Geo` endpoints for each rack in IMS and data racks.
- In `SMF-Ops-Center` CLI, you can configure `GTPC-EP Geo` endpoints for each rack in IMS and DATA racks.

Note The optimization of GTPC messages can be applied to all four instances or a subset of instances of GTPC endpoints within these racks. A new element is added under the GTPC endpoints to configure a list of IP addresses where SMF and cnSGW-C service pods can route the GTPv2 messages over the IPC interface.

Upgrading and Enabling Inter-rack GTPC IPC

This section describes how to upgrade and enable the inter-rack GTPC IPC.

Before you configure, upgrade, and enable the GTPC IPC optimization on cnSGW-C and SMF interfaces, you must perform the following:

- The inter-rack routing must be applied to the branched core network.
- More configuration parameters such as GTPC, cnSGW-C, and SMF are enabled, when extra routes are added for supporting a new routable network across rack servers.

Note The following are configuration-related points:

- The SGW service pods for the S5 egress MBR request are used for IPC messages toward PGWc and SMF IP in the list.
- The SMF service pods for S5 CBR, UBR, and DBR requests are used for IPC messages toward the cnSGW-C in the list.
- IPC messages reuse the N3 or the T3 configuration for the respective interfaces to retry messages, whenever the timeout in the peer node occurs.

Call Flows

This section describes the key call flows for this feature.

cnSGW-C GTPC Optimization with Inter-rack IPC Call Flow

This section describes the cnSGW-C GTPC Optimization with Inter-rack IPC call flow.

Figure 103: cnSGW-C GTPC Optimization with Inter-rack IPC Call Flow

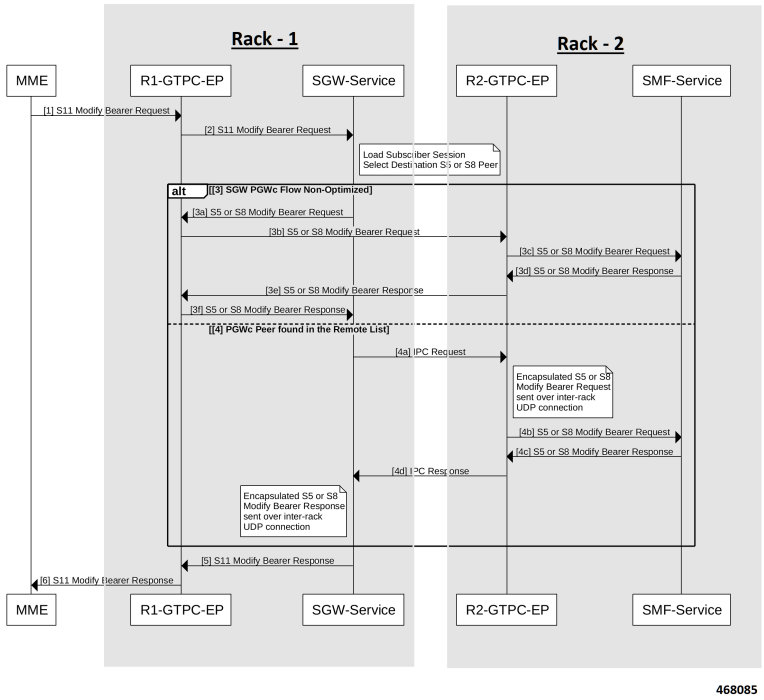


Table 185: cnSGW-C GTPC Optimization with Inter-rack IPC Call Flow Description

Step	Description
1	MME sends the S11 Modify Bearer Request to R1-GTPC-EP.
2	R1-GTPC-EP sends the S11 Modify Bearer Request to SGW-service. **Note** The SGW-service section performs the following: • Loads the subscriber session. • Selects the destination as the S5 or the S8 peer.
3	The following are sub-steps in the Alt SGW PGWc flow non-optimized scenario.
3a	SGW-service sends the S5 or the S8 Modify Bearer Request to R1-GTPC-EP.
3b	R1-GTPC-EP sends the S5 or the S8 Modify Bearer Request to R2-GTPC-EP.

Step	Description
3c	R2-GTPC-EP sends the S5 or the S8 Modify Bearer Request to SMF-service.
3d	SMF-service processes and sends the S5 or the S8 Modify Bearer Response to R2-GTPC-EP.
3e	R2-GTPC-EP sends the Modify Bearer Response to R1-GTPC-EP.
3f	R1-GTPC-EP sends the Modify Bearer Response to SGW-service.
4	Alternatively, the PGWc peer scenario is available in the remote list. The following are sub-steps in this scenario.
4a	SGW-service sends the IPC Request to R2-GTPC-EP. **Note** The R2-GTPC-EP section performs the following: • Encapsulates the S5 or the S8 Modify Bearer Request. • Sends it over the inter-rack UDP connection.
4b	R2-GTPC-EP sends the S5 or the S8 Modify Bearer Request to SMF-service.
4c	SMF-service processes and sends the S5 or the S8 Modify Bearer Response to R2-GTPC-EP.
4d	R2-GTPC-EP sends the IPC Response to SGW-service. **Note** The SGW-service section performs the following: • Encapsulates the S5 or the S8 Modify Bearer Response. • Sends it over the inter-rack UDP connection.
5	SGW-service sends the Modify Bearer Response to R1-GTPC-EP.
6	R1-GTPC-EP sends the Modify Bearer Response to MME.

SMF and PGWc GTPC Optimization with Inter-rack IPC Call Flow

This section describes the SMF and PGWc GTPC Optimization with Inter-rack IPC call flow.

Figure 104: SMF and PGWc GTPC Optimization with Inter-rack IPC Call Flow

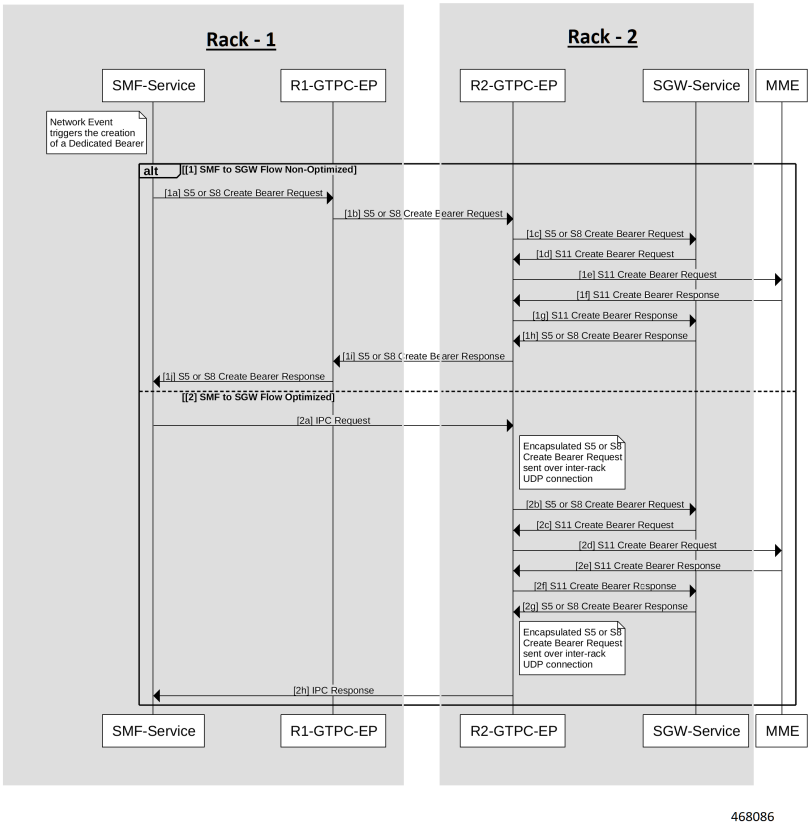


Table 186: SMF and PGWc GTPC Optimization with Inter-rack IPC Call Flow Description

Step	Description
1	The following are sub-steps in the Alt SMF to SGW flow non-optimized scenario. **Note** The SMF-service section performs the following: • Triggers the networking event. • Creates the resolute bearer.
1a	SMF-service sends the S5 or the S8 Create Bearer Request to R1-GTPC-EP.
1b	R1-GTPC-EP sends the S5 or the S8 Create Bearer Request to R2-GTPC-EP.
1c	R2-GTPC-EP sends the S5 or the S8 Create Bearer Request to SGW-service.
1d	SGW-service processes and sends the S11 Create Bearer Request to R2-GTPC-EP.
1e	R2-GTPC-EP sends the S11 Create Bearer Request to MME.

Step	Description
1f	MME processes and sends the S11 Create Bearer Response to R2-GTPC-EP.
1g	R2-GTPC-EP sends the S11 Create Bearer Response to SGW-service.
1h	SGW-service processes and sends the S5 or the S8 Create Bearer Response to R2-GTPC-EP.
1i	R2-GTPC-EP sends the S5 or the S8 Create Bearer Response to R1-GTPC-EP.
1j	R1-GTPC-EP sends the S5 or the S8 Create Bearer Response to SMF-service.
2	The following are sub-steps in the SMF to SGW flow-optimized scenario.
2a	SMF-service sends the IPC request to R2-GTPC-EP. **Note** The R2-GTPC-EP section performs the following: • Encapsulates the S5 or the S8 Create Bearer Request. • Sends it over the inter-rack UDP connection.
2b	R2-GTPC-EP sends the S5 or the S8 Create Bearer Request to SGW-service.
2c	SGW-service processes and sends the S11 Create Bearer Request to R2-GTPC-EP.
2d	R2-GTPC-EP sends the S11 Create Bearer Request to MME.
2e	MME processes and sends the S11 Create Bearer Response to R2-GTPC-EP.
2f	R2-GTPC-EP sends the S11 Create Bearer Response to SGW-service.
2g	SGW-service processes and sends the S5 or the S8 Create Bearer Response to R2-GTPC-EP. **Note** The R2-GTPC-EP section performs the following: • Encapsulates the S5 or the S8 Create Bearer Request. • Sends it over the inter-rack UDP connection.
2h	R2-GTPC-EP processes and sends the IPC Response to SMF-service.

Feature Configuration

To configure this feature, use the following configuration:

```
config
instance instance_id 1
    endpoint endpoint_name
        interface gtp-inter-rack gtp_inter_rack_name
            vip-ip vip_ip_address vip-port vip_port_address vip-interface
vip_interface_address
```

```
          gtpc-ipc gtpc_ipc_name
          gtp-peer-entry gtp_peer_entry_address port port_address
remote-gtp-peer-list remote_gtp_peer_list_addresses
          end
```

Note In the preceding new configuration example, the following are the enhanced scenarios:

- The cnSGW-C service pod for the S5 egress MBR request will be using IPC messages toward the PGWc or the SMF IP in the preceding list.
- Similarly, the SMF service pod for S5 CBR, UBR, and DBR requests are using IPC messages toward cnSGW in the preceding list.
- IPC messages reuse the N3 or T3 configuration for the respective interface to retry messages, when there is a timeout in the peer node.

NOTES:

- **instance** *instance_id 1*—Specify the instance ID.
- **endpoint** *endpoint_name*—Specify the endpoint name.
- **gtp-inter-rack** *gtp_inter_rack_name*—Specify the interface name. Specify the **gtp-inter-rack** name, you want to select. It is a new interface, added for cross-rack routing.
- **vip-ip** *vip_ip_address* **vip-port** *vip_port_address* **vip-interface** *vip_interface_address*—Specify the addresses for **vip-ip**, which is a GTP IPC endpoint server IP, **vip-port**, which is a GTP IPC endpoint server listening port, and **vip-interface**, which is a GTP IPC endpoint server interface VLAN.
- **gtpc-ipc** *gtpc_ipc_name*—Specify the interface name. Specify the **gtp-ipc** name, you want to select.
- **gtp-peer-entry** *gtp_peer_entry_address* **port** *port_address* **remote-gtp-peer-list** *remote_gtp_peer_list_addresses*—Specify the addresses for the list of **gtp-peer-entry**, which is a remote GTP IPC peer IP configured on other racks or instances (multiple rows), **port**, which is a remote GTP IPC peer port, and **remote-gtp-peer-list**, which is a list of S5 and S5e remote GTP peers endpoints on a rack or instances corresponding to the **gtp-peer-entry**.

Configuration Example

The following is an example configuration.

```
config
instance instance-id 1
 endpoint GTP
   interface gtp-inter-rack
     vip-ip 209.165.202.130 vip-port 9084 vip-interface bd2.gtpe.2101
     gtpc-ipc
     gtp-peer-entry 209.165.202.131 port 9084 remote-gtp-peer-list [ 209.165.202.140
209.165.202.141 ]
     end
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

KPI Support

The following statistics are supported for the GTPC IPC Cross-rack Support feature.

1. KPI Name: **udp_rpc_request_total**

   The following table lists **udp_rpc_request_total** KPI details.

Description	Functionality	Label Names	Possible Values
This KPI displays the total number of UDP client endpoint request messages.	The functionality output is inter-rack RPC requests total.	MessageName	gptv2 message types like CBR, UBR, MBR
		retry	Is attempt a retry
		status	Status of the request message, which is a success or a failure
		status_code	0/1/2

2. KPI Name: **udp_rpc_response_total**

   The following table lists **udp_rpc_response_total** KPI details.

Description	Functionality	Label Names	Possible Values
This KPI displays the total number of UDP client endpoint response messages.	The functionality output is inter-rack RPC response total.	MessageName	gptv2 message types like CBR, UBR, MBR
		status	Status of the response message, which is a success or a failure
		status_code	0/1/2

Note The current implementation supports KPIs only on the client-side, as they reside on service pods, where KPIs can be enabled without impacting performance.

Interservice Pod Communication

Feature Description

When the IMS PDN sgw-service and smf-service selected for a subscriber are on the same cluster and same RACK, the following message flow occurs when sgw-service sends a message to smf-service:

- The message is sent from S5e gtpc-ep interface to network interface.
- The message returns to the S5 interface from gtpc-ep to smf-service.

For the subscribers that are collocated, the communication happens between the sgw-service and the smf-service. This approach reduces the processing load on the gtpc-ep.



Note A direct communication between sgw-service and smf-service is not supported to transfer messages on monitor protocol and monitor subscriber.

How it Works

This section describes how this feature works.

The sgw-service communicates with smf-service for processing the following requests:

- Create Session Request
- Modify Bearer Request
- Delete Session Request

The smf-service communicates with sgw-service for processing the following requests:

- Create Bearer Request
- Update Bearer Request
- Delete Bearer Request

The sgw-service sends the Modify Bearer Command and Delete Bearer Command messages to SMF through gtpc-ep. If the Update Bearer Request and Delete Bearer Request is triggered, the command messages are sent to sgw-service through gtpc-ep.

Call Flows

This section describes the key call flows for this feature.

Collapsed Call Attach with SGW-Service to SMF-Service Configuration Call Flow

This section describes the Collapsed Call Attach with SGW-Service to SMF-Service Configuration call flow.

Figure 105: Collapsed Call Attach with SGW-Service to SMF-Service Configuration Call Flow

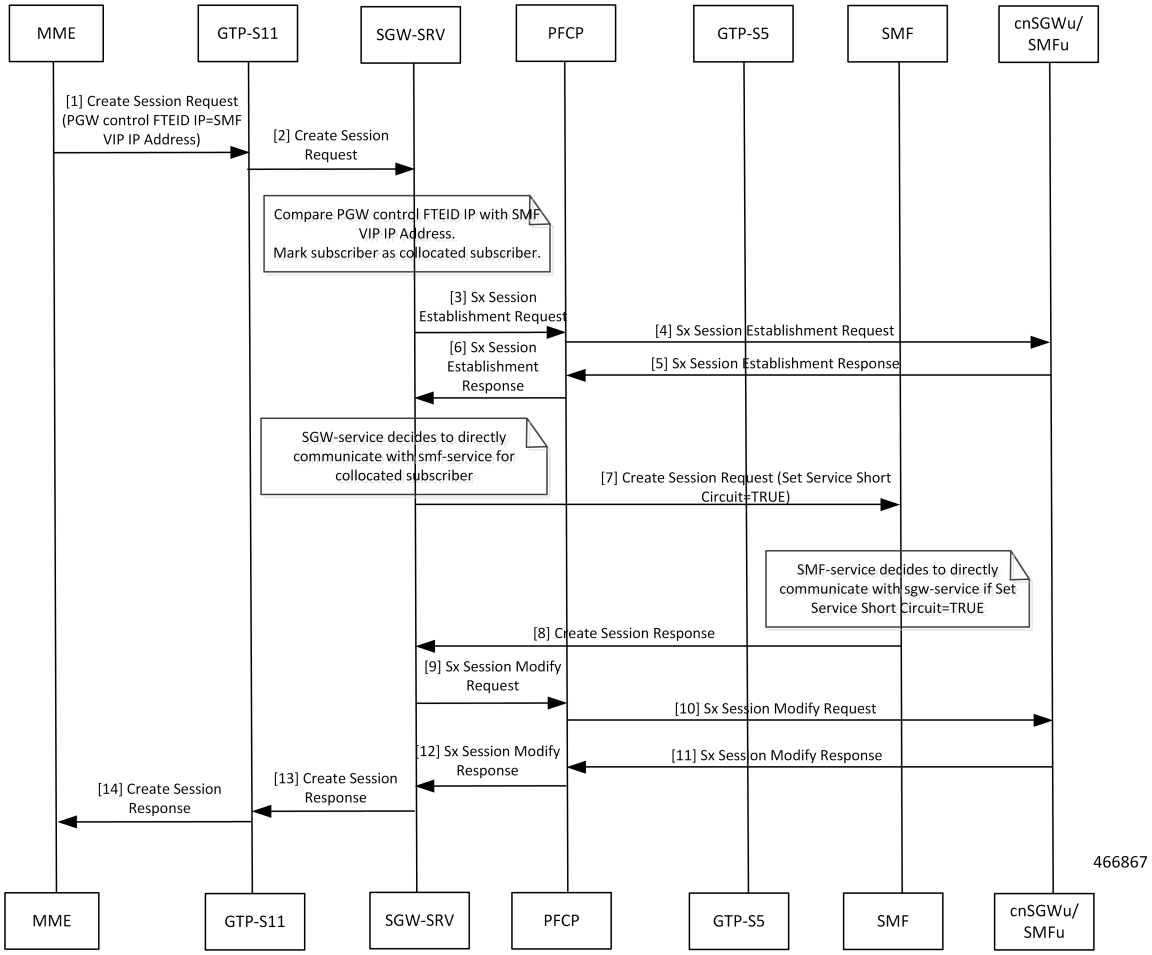


Table 187: Collapsed Call Attach with SGW-Service to SMF-Service Configuration Call Flow Description

Step	Description
1	MME sends the Create Session Request to GTP-S11.
2	GTP-S11 sends the Create Session Request to SGW-SRV.
3	SGW-SRV sends the Sx Session Establishment Request to PFCP.
4	PFCP sends the Sx Session Establishment Request to cnSGWu/SMFu.
5	cnSGWu/SMFu sends the Sx Session Establishment Response to PFCP.
6	PFCP sends the Sx Session Establishment Response to SGW-SRV.
7	SGW-SRV sends the Create Session Request to SMF. The Service Short Circuit is set to TRUE.
8	SMF sends the Create Session Response to SGW-SRV.
9	SGW-SRV sends the Sx Session Modify Request to PFCP.

Step	Description
10	PFCP sends the Sx Session Modify Request to cnSGWu/SMFu.
11	cnSGWu/SMFu sends the Sx Session Modify Response to PFCP.
12	PFCP sends the Sx Session Modify Response to SGW-SRV.
13	SGW-SRV sends the Create Session Response to GTP-S11.
14	GTP-S11 sends the Create Session Response to MME. For a collocated subscriber, when SGW-SRV receives: • Modify Bearer Request and Delete Session Request from the MME, the SGW-SRV communicates these messages to the SMF service. • Create Bearer Request, Update Bearer Request, and Delete Bearer Request from the SMF, the SGW-SRV communicates the response messages for these request messages to the SMF service.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Statistics Support

To check for messages that are directly communicated to SMF service, add svc_to_svc field in sgw_service_stats query as shown below:

Query: sum(irate(sgw_service_stats{status=~"attempted"}[30s])) by (sgw_procedure_type,status, interface, svc_to_svc)

Legend: {{interface}} -> {{sgw_procedure_type}} | {{svc_to_svc}} | {{status}}

MBR Call Flow Optimization

Feature Description

cnSGW-C supports optimization of Modify Bearer Request and Modify Bearer Response (MBR) call flows to reduce I/O operation, reduce transaction wait time, and improve performance in multi-PDN scenarios.

How it Works

This section describes how this feature works.

The following functions explain the optimization of MBR call flows:

- To reduce I/O operations, cnSGW-C combines all Modify Bearer Requests towards SGW-service into a single GRPC call, and Sx Modify Requests from SGW-service pod to protocol pod in a single GRPC call.

- To reduce transaction wait time in GTPC-EP, cnSGW-C sends Modify Bearer Response immediately from GTPC-EP (except last MBR) after receiving Modify Bearer Request.

  GTPC-EP combines all Modify Bearer Requests in a single Modify Bearer Request List message and sends to SGW-service.

- SGW-service combines all Modify Bearer Responses into a single Modify Bearer Response List message and sends to GTPC-EP.

  SGW-service combines all Sx Modify Requests towards UPF into a single Sx Modify Request List message and sends to protocol pod. The protocol pod sends individual Sx Modify Requests to UPF.

- The protocol pod waits for all Sx Modify Responses from UPF and combines them into a single Sx Modify Response List and sends it to SGW-service.

- In non-merged mode, UDP proxy maintains the local TEID and remote TEID cache information. In merged mode, GTPC-EP maintains the local TEID and remote TEID cache information.

- If GTPC-EP does not find the TEID cache entry for the received Modify Bearer Request, the Modify Bearer Request will be forwarded to the SGW-service immediately.

  If all expected Modify Bearer Requests are not received within the MBR cache expiry, only the Modify Bearer Requests that are received will be sent to the SGW-service.

Call Flows

This section describes the key call flows for this feature.

Idle-Active Transition with Intra MME HO Call Flow

This section describes the current call flow with intra MME handover (NSA MBR) for moving from eNodeB to gNodeB.

Figure 106: Idle-Active Transition with Intra MME HO Call Flow

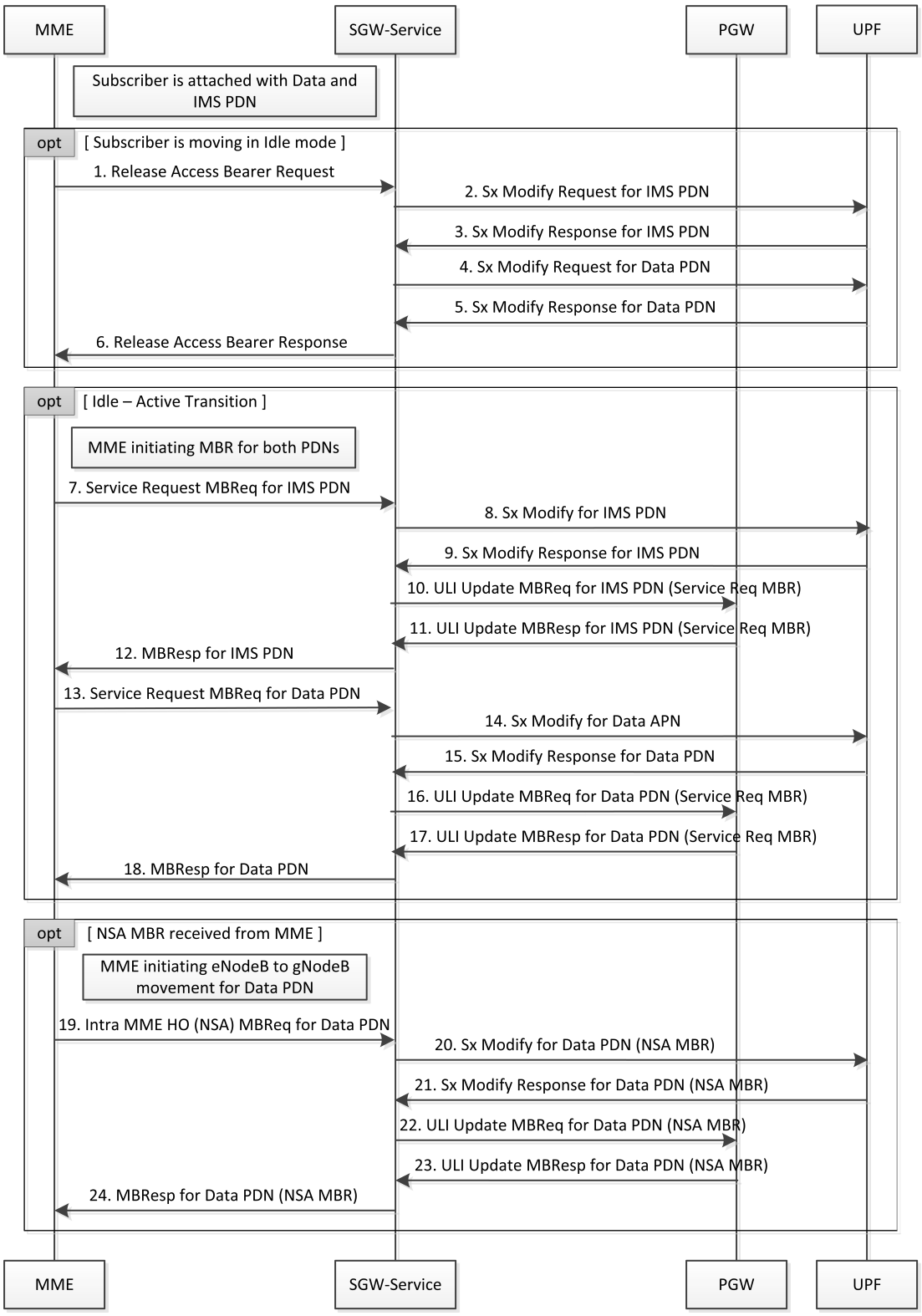

MME
SGW-Service
PGW
UPF
Subscriber is attached with Data and IMS PDN
opt
[Subscriber is moving in Idle mode]
1. Release Access Bearer Request
2. Sx Modify Request for IMS PDN
3. Sx Modify Response for IMS PDN
4. Sx Modify Request for Data PDN
5. Sx Modify Response for Data PDN
6. Release Access Bearer Response
opt
[Idle – Active Transition]
MME initiating MBR for both PDNs
7. Service Request MBReq for IMS PDN
8. Sx Modify for IMS PDN
9. Sx Modify Response for IMS PDN
10. ULI Update MBReq for IMS PDN (Service Req MBR)
11. ULI Update MBResp for IMS PDN (Service Req MBR)
12. MBResp for IMS PDN
13. Service Request MBReq for Data PDN
14. Sx Modify for Data APN
15. Sx Modify Response for Data PDN
16. ULI Update MBReq for Data PDN (Service Req MBR)
17. ULI Update MBResp for Data PDN (Service Req MBR)
18. MBResp for Data PDN
opt
[NSA MBR received from MME]
MME initiating eNodeB to gNodeB movement for Data PDN
19. Intra MME HO (NSA) MBReq for Data PDN
20. Sx Modify for Data PDN (NSA MBR)
21. Sx Modify Response for Data PDN (NSA MBR)
22. ULI Update MBReq for Data PDN (NSA MBR)
23. ULI Update MBResp for Data PDN (NSA MBR)
24. MBResp for Data PDN (NSA MBR)
MME
SGW-Service
PGW
UPF
466863

Table 188: Idle-Active Transition with Intra MME HO Call Flow Description

Step	Description
1	MME sends Release Access Bearer (RAB) request to SGW-service.
2	SGW-service sends Sx Modify Request for IMS PDN to UPF.
3	UPF sends Sx Modify Response for IMS PDN to SGW-service.
4	SGW-service sends Sx Modify Request for Data PDN to UPF.
5	UPF sends Sx Modify Response for Data PDN to SGW-service.
6	SGW-service sends the Release Access Bearer Response message to MME.
7	MME sends Service Request MBRequest for IMS PDN to SGW-service.
8	SGW-service sends Sx Modify Request for IMS PDN to UPF.
9	UPF sends Sx Modify Response for IMS PDN to SGW-service.
10	SGW-service sends ULI Update MBRequest for IMS PDN (Service Request MBR) to PGW.
11	PGW sends ULI Update MBResponse for IMS PDN (Service Request MBR) to SGW-service.
12	SGW-service sends MBResponse for IMS PDN to MME.
13	MME sends Service Request MBRequest for Data PDN to SGW-service.
14	SGW-service sends Sx Modify Request for Data PDN to UPF.
15	UPF sends Sx Modify Response for Data PDN to SGW-service.
16	SGW-service sends ULI Update MBRequest for Data PDN (Service Request MBR) to PGW.
17	PGW sends ULI Update MBResponse for Data PDN (Service Request MBR) to SGW-service.
18	SGW-service sends MBResponse for Data PDN to MME.
19	MME sends intra MME HO (NSA) MBRequest for Data PDN to SGW-service.
20	SGW-service sends Sx Modify Request for Data PDN (NSA MBR) to UPF.
21	UPF sends Sx Modify Response for Data PDN (NSA MBR) to SGW-service.
22	SGW-service sends ULI Update MBRequest for Data PDN (NSA MBR) to UPF.
23	UPF sends ULI Update MBResponse for Data PDN to SGW-service.
24	SGW-service sends MBResponse for Data PDN (NSA MBR) to MME.

MBR Optimization Call Flow

This section describes the high-level MBR Optimization call flow for idle↔active transition.

Figure 107: MBR Optimization Call Flow

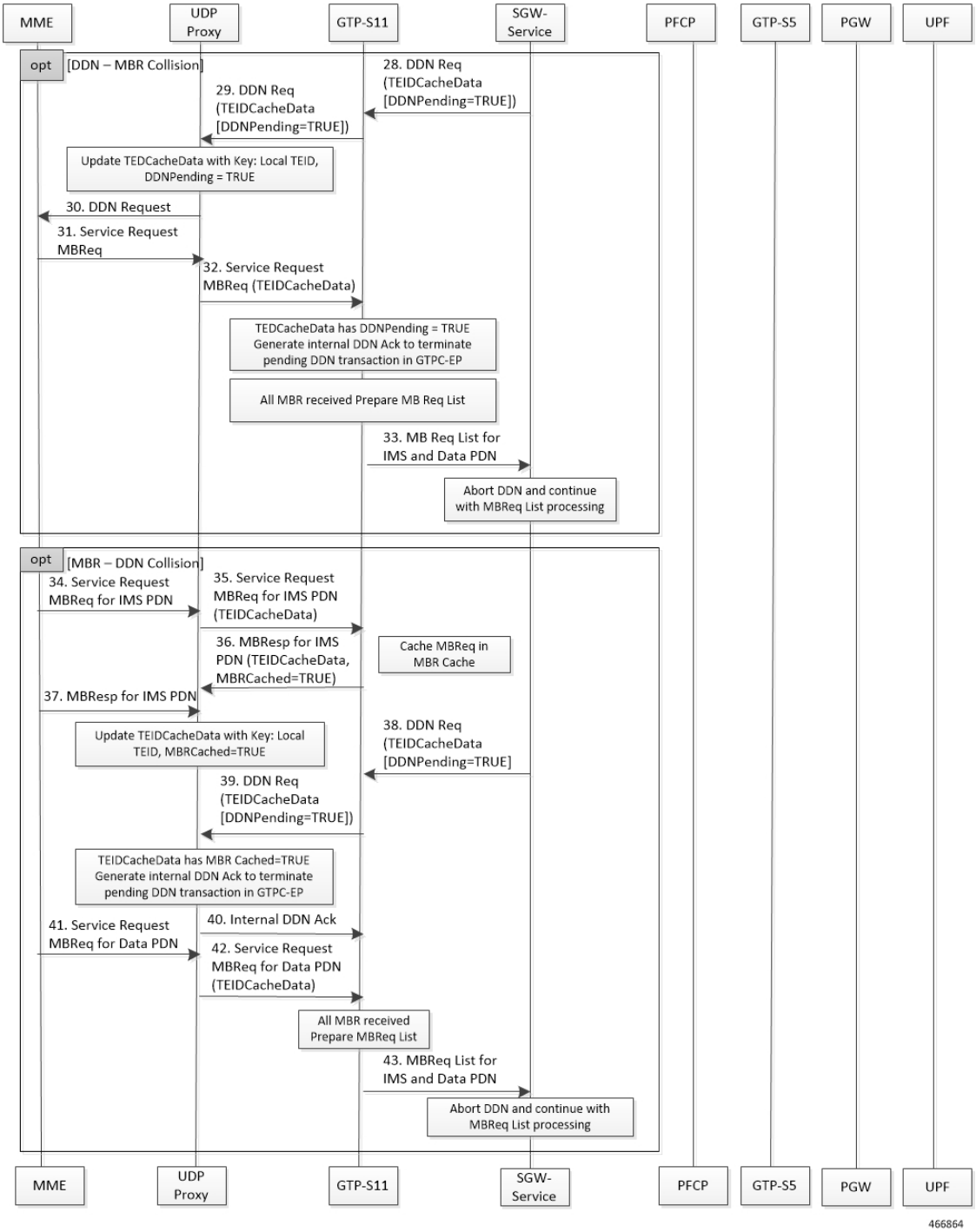


Table 189: MBR Optimization Call Flow Description

Step	Description
1	MME sends Release Access Bearer (RAB) request to UDP proxy.
2	UDP Proxy forwards RAB Request to GTP-S11.
3	GTP-S11 forwards RAB Request to SGW-service.
4	SGW-service sends Sx Modify List for all PDNs to PFCP.
5	SGW-service prepones RAB Response.
6	GTP-S11 sends RAB Response to UDP Proxy.
7	UDP Proxy sends RAB Response to MME.
8	PFCP sends and receives Sx Modify for IMS APN to and from UPF.
9	PFCP sends and receives Sx Modify for Data APN to and from UPF.
10	PFCP sends Sx Modify Response List for all PDNs to SGW-service.
11	MME sends Service Request MBRequest for IMS PDN to UDP Proxy.
12	UDP Proxy forwards Service Request MBRequest for IMS PDN with TEID cache data to GTP-S11.
13	GTP-S11 sends MBResponse for IMS PDN to UDP Proxy.
14	UDP Proxy forwards MBResponse for IMS PDN to MME.
15	MME sends Service Request MBRequest for Data PDN to UDP Proxy.
16	UDP Proxy sends Service Request MBRequest for Data PDN with TEID cache data to GTP-S11.
17	GTP-S11 sends MBRequest List for IMS and Data PDNs to SGW-service.
18	SGW-service sends Sx Modify List for IMS and Data PDN to PFCP.
19	PFCP sends and receives Sx Modify for IMS PDN to and from UPF.
20	PFCP sends and receives Sx Modify for Data PDN to and from UPF.
21	PFCP sends Sx Modify Response List for IMS and Data PDN to SGW-service.
22	SGW-service sends ULI Update MBRequest List for IMS and Data PDN (Async Notification) to GTP-S5.
23	SGW-service sends MBResponse List for Data PDN to GTP-S11.
24	GTP-S11 sends MBResponse for Data PDN to UDP Proxy.
25	UDP Proxy forwards MBResponse for Data PDN to MME.
26	GTP-S5 sends and receives ULI Update MBRequest for IMS PDN to and from PGW.

Step	Description
27	GTP-S5 sends and receives ULI Update MBRequest for Data PDN to and from PGW.
28	SGW-service sends DDN Request to GTP-S11.
29	S-11 forwards DDN Request to UDP Proxy.
30	UDP Proxy forwards DDN Request to MME.
31	MME sends Service Request MBRequest to UDP Proxy.
32	UDP Proxy sends Service Request MBRequest with TEIDCacheData to GTP-S11.
33	GTP-S11 sends MBResponse List for IMS and Data PDNs to SGW-service.
34	MME sends Service Request MBRequest for IMS PDN to UDP Proxy.
35	UDP Proxy Service Request MBRequest for IMS PDN with TEID cache data to GTP-S11.
36	GTP-S11 sends MBResponse for IMS PDN to UDP Proxy.
37	UDP Proxy sends MBResponse for IMS PDN to MME.
38	SGW-service sends DDN Request to GTP-S11.
39	GTP-S11 forwards DDN Request to UDP Proxy.
40	UDP Proxy sends internal DDN acknowledgement to GTP-S11.
41	MME sends Service Request MBRequest for Data PDN to UDP Proxy.
42	MME sends Service Request MBRequest for Data PDN with TEID cache data to GTP-S11.
43	GTP-S11 sends MBRequest List for IMS and Data PDNs to SGW-service.

Feature Configuration

To configure this feature, use the following sample configuration:

```
config
   instance istance-id instance_id
      endpoint endpoint_name
         mbr-optimization [ enable { false | true } | mbr-cache-expiry
mbr_cache | teid-cache-expiry teid_cache ]
         exit
      exit
   exit
```

NOTES:

- **mbr-optimization [enable { false | true } | mbr-cache-expiry** *mbr_cache* **| teid-cache-expiry** *teid_cache* **]**: Specify the MBR optimization configuration.
  - **enable { false | true }**: Enable or disable MBR optimization. Default: Disabled.

- **mbr-cache-expiry** *mbr_cache* : Specify the MBR cache expiry time interval in milliseconds, as an integer from 1 millisecond to 6 seconds. Default: 50 milliseconds.

  Note that the value of **mbr-cache-expiry** can be changed during the runtime.

- **teid-cache-expiry** *teid_cache* : Specify the TEID cache expiry time interval in milliseconds, as an integer from 1000 milliseconds to 1 hour. Default: 120000 milliseconds.

Configuration Example

The following is an example configuration.

```
config
   instance instance-id 1
      endpoint gtp
         mbr-optimization enable true mbr-cache-expiry 60 teid-cache-expiry 180000
         end
```

Configuration Verification

To verify the configuration:

```
show running-config instance instance-id 1 endpoint gtp
instance instance-id 1
endpoint gtp
  replicas                1
  nodes                   1
  mbr-optimization
   enable            true
   teid-cache-expiry 180000
   mbr-cache-expiry  60
  exit
  enable-cpu-optimization true
  ……
```

OAM Support

This section describes operations, administration, and maintenance support for the MBR Optimization feature.

Bulk Statistics Support

The following statistics are supported for the MBR Optimization feature.

The SGW-service supports the *service_request_list* procedure type to handle Modify Bearer Request list from GTPC-EP.

Query: `sum(rate(sgw_service_stats{namespace=~"$namespace", interface="interface_sgw_ingress",sgw_procedure_type="service_request_list", status="attempted"}[1m]))`

Legend: `IDLE -> ACTIVE (List)`

The MBR short circuit statistics are enhanced to capture statistics for Modify Bearer Request list sent to SGW-service.

Query: `sum(irate(gtpc_msg_short_circuit_stats{namespace=~"$namespace"}[60s])) by (gtpc_msg_type, gtpc_short_circuit_category)`

Legend: `{{gtpc_msg_type}}, {{gtpc_short_circuit_category}}`

This feature supports the following statistics:

- RxModifyBearerReq, SendSCMBResp—The number of early MB responses sent from GTPC-EP.
- RxModifyBearerReq, SendMBReqListToSrv—The number of MB request lists sent to SGW-service.
- RxModifyBearerReq, SendMBReqToSrv—The number of MB requests sent to SGW-service.
- RxModifyBearerReq, MBREventExpired—The number of MB requests expired in MBR cache.

Maintenance Mode

Feature Description

The maintenance mode feature allows a cluster in the GR setup to undergo an in-service upgrade (rolling upgrade) without any service disruption. Maintenance mode works with routing the traffic to the mated cluster to perform the responsibility of the source cluster.

How it Works

When the maintenance mode flag is set to true, the cluster role changes and GR is triggered for the rack. The standby cluster takes over the responsibility of the cluster that is in the maintainence mode. During this period, the monitoring threads check the runtime value of the flag and pause the execution when the maintenance mode flag is set to true. By default, for fresh installation, the flag is set to false.

Both, the source and standby clusters (racks) can be under the maintenance mode at the same time. You can enable the maintenance mode for the rack server regardless of its state.

You can push the system to the maintenance mode when the maintenance procedure is in-progress for the mated cluster. Before you start the maintenance activity, set the `geo maintenance mode` flag value to true. When the maintenance is complete, reset the flag to false after confirming the health of the system.

For information on how to configure the maintenance mode flag, see Enabling or Disabling Maintenance Mode, on page 494.

When the maintenance mode is enabled:

- Automated GR-switchover such as pod monitoring, BFD link monitoring from the rack server is not supported.
- Only CLI-based GR-switchover is supported from the rack (with maintenance mode enabled) to the partner rack.
- GR-switchover, including CLI-based, is not supported from the partner rack to the rack where the maintenance mode is enabled.
- If both partner racks are in the maintenance mode, GR-switchover is not supported.
- All the monitoring activities are paused.
- The mated cluster cannot trigger the failover when it detects the local failure.

- Replication activities continue on the cluster.
- Maintenance mode doesn't implicitly change the instance roles of the site. However, role change is possible using the **geo switch-role role** CLI command.

  GR trigger is not supported towards and from the cluster that is under maintenance. Only CLI-based failover is supported from the cluster under the maintenance.

Limitations

This feature has the following limitation in this release:

The maintenance mode feature does not overwrite the multicompute failure switchover case. However, the multicompute failure switchover scenario is supported when the partner rack is also in maintenance mode.



Note The multicompute failure is a switchover case that occurs when multiple servers in a rack fail causing the partner rack to handle the traffic.

Enabling or Disabling Maintenance Mode

To enable or disable this feature, use the following command:

`geo maintenance mode { true | false }`

NOTES:

geo maintenance mode { true | false }—Specify to enable or disable the maintenance mode. To enable the maintenance mode, specify true. If the maintenance mode flag is set to true, the cluster role changes and GR is triggered for the rack.

The value for the maintenance mode is stored in the `.etcd` file.

Enabling or Disabling Maintenance Mode Example

The following is an example of the command:

```
geo maintenance mode true
```

```
geo maintenance mode false
```

Verifying the Maintenance Mode State

To verify the maintenance mode state:

```
show geo maintenance mode
result "geo maintenance mode is disabled"
```

Partial CDL Update for Idle-Active Call Flow

Feature Description

cnSGW-C supports partial CDL update of the subscriber data. Partial CDL update helps in saving the CPU requirement in the CDL pod for processing the subscriber data.

With each transition of the subscriber from Idle-to-Active state and Active-to-Idle state, cnSGW-C updates only the following fields:

- eNodeB FTEID
- Subscriber state
- Bearer state

How it Works

This section describes how this feature works.

cnSGW-C stores the update bearer information in the Flags database in the CDL.

cnSGW-C uses partial CDL update when the subscriber moves from:

- Active state to Idle state on receiving Release Access Bearer Request
- Idle state to Active state on receiving Modify Bearer Request

With partial CDL update, the session-state-flag displays the following value in the **cdl show sessions summary slice-name <n>** CLI output:

- sgw_active: when the session is Active
- sgw_inactive: when the session is Idle

The following is a sample output for an Active session:

```
cdl show sessions summary slice-name 1
message params: {session-summary cli session {0 100  0 [] 0 0 false 4096 [] []} 1}
session {
    primary-key 2#/#imsi-123456789012348
    unique-keys [ "2#/#16777229" ]
    non-unique-keys [ "2#/#id-index:1:0:32768" "2#/#id-value:16777229"
    "2#/#imsi:imsi-123456789012348" "2#/#msisdn:msisdn-223310101010101"
    "2#/#imei:imei-123456786666660" "2#/#upf:209.165.201.1"
    "2#/#upfEpKey:209.165.201.1:209.165.201.30" "2#/#s5s8Ipv4:209.165.202.129"
"2#/#s11Ipv4:209.165.201.1"
    "2#/#namespace:sgw" ]
    flags [ byte-flag1:00:13:03:53:00:00:06:85:0A:01:01:1B session-state-flag:sgw_active ]
    map-id 1
    instance-id 1
    app-instance-id 1
    version 1
    create-time 2022-01-20 11:37:15.181259564 +0000 UTC
    last-updated-time 2022-01-20 11:37:15.703032336 +0000 UTC
    purge-on-eval false
```

```
    next-eval-time 2022-01-27 11:37:26 +0000 UTC
    session-types [ SGW:rat_type:EUTRAN ]
    data-size 925
```

The following is a sample output for an Idle session:

```
cdl show sessions summary slice-name 1
message params: {session-summary cli session {0 100  0 [] 0 0 false 4096 [] []} 1}
session {
    primary-key 2#/#imsi-123456789012348
    unique-keys [ "2#/#16777229" ]
    non-unique-keys [ "2#/#id-index:1:0:32768" "2#/#id-value:16777229"
    "2#/#imsi:imsi-123456789012348" "2#/#msisdn:msisdn-223310101010101"
    "2#/#imei:imei-123456786666660" "2#/#upf:209.165.201.1"
"2#/#upfEpKey:209.165.201.1:209.165.201.30"
    "2#/#s5s8Ipv4:209.165.202.129" "2#/#s11Ipv4:209.165.201.1" "2#/#namespace:sgw" ]
    flags [ byte-flag1:00:25:00:55:00:65 session-state-flag:sgw_inactive ]
    map-id 1
    instance-id 1
    app-instance-id 1
    version 3
    create-time 2022-01-20 11:37:15.181259564 +0000 UTC
    last-updated-time 2022-01-20 11:37:18.102852792 +0000 UTC
    purge-on-eval false
    next-eval-time 2022-01-27 11:37:28 +0000 UTC
    session-types [ SGW:rat_type:EUTRAN ]
    data-size 1644
```

Limitations

cnSGW-C doesn't support partial CDL update for IPv6 TEID.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   cdl
      datastore datastore_session_name
         slot metrics report-idle-session-type { true | false }
         end
```

NOTES:

- **slot metrics report-idle-session-type { true | false }**—Enable or disable Idle or Active session count in CDL db_records_total.

Configuration Example

The following is an example configuration.

```
config
   cdl
      datastore session
         slot metrics report-idle-session-type true
         end
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the Partial CDL Update Idle-Active Call Flow feature.

Grafana Query to find cnSGW-C Idle and Active Session count:

```
SGW ACTIVE COUNT :-
(db_records_total{namespace=~"$namespace",session_type=~"sgw_active"})
SGW_IDLE COUNT :
(db_records_total{namespace=~"$namespace",session_type=~"sgw_inactive"})
```

PFCP Session Report with DLDR Throttling Support

Feature Description

In a live network deployment, due to some external events, all or most of the idle sessions become active at the same time. When idle sessions become active, the UPF sends the session report request with the report type DLDR to the cnSGW-C.

When the cnSGW-C receives the session report with the report type as DLDR, the cnSGW-C sends the DDN message to page UE. To turn the UE active, the MME initiates the paging procedure for the UE. If paging is successful, the MME initiates the service request Modify Bearer Request. On delivering data to the UE, the UE initiates the Release Access Bearer Request and turns idle. This call flow increases an overall load on the system. When the entire call flow occurs for all subscribers in a short time, there's a huge process overhead on the system.

The PFCP Session Report with the DLDR Throttling Support feature enables the cnSGW-C to limit the number of session report requests that enter the system to prevent the process overload on the system.



Important This functionality will also throttle emergency calls.

How it Works

This section describes how this feature works.

The cnSGW-C uses the app-infra feature for SBA overload control to throttle the incoming messages.

The system takes appropriate action for the incoming messages that are based on the interface-level threshold configuration. The incoming messages get added to different queues and they are processed based on the message-level priority configuration.

For more information on Overload Support, see the *SMF Overload Support* chapter, in the *UCC 5G Session Management Function Configuration and Administration Guide*.

You can exclude session reports for emergency call, voice calls, and empty calls from throttling.

To exclude these session reports, configure the ddn delay-exclude-arplist configuration in profile sgw. If the session report is received for one of the configured ARPs, the cnSGW-C omits that session report from the session report throttling.

Feature Configuration

To configure this feature, use the following configuration:

```
config
   instance instance-id instance_id
      endpoint pfcp
         interface sxa
            overload-control msg-type session-report
            rate-limit rate_limit
            queue-size queue_size
            reject-threshold reject_threshold
            pending-request pending_request
            exit
config
   profile sgw sgw_name
      ddn delay-exclude-arplist number_priorities
      end
```

NOTES:

- **overload-control msg-type session-report**—Configure the virtual message specifications for interface overload.
- **rate-limit** *rate_limit*—Specify the rate limit for the virtual queue.
- **queue-size** *queue_size*—Specify the packet count or capacity of each virtual queue.
- **reject-threshold** *reject_threshold*—Specify the limit to reject incoming messages when this threshold percentage of pending requests is reached.
- **pending-request** *pending_request*—Specify the pending requests count in the virtual queue.
- **ddn delay-exclude-arplist** *number_priorities*—Specify the priority-level for allocation and retention priorities [1-15] that must be excluded from delaying the DDN/Session report throttling.

Important

- Both GRs instances must have the same rate limiter configuration.
- Session report throttling must be configured in such a way that the waiting time in the queue should not exceed transaction SLA (pending-request/rate-limit < Txn SLA (in seconds)).
- After adding this configuration, the Protocol pod will be restarted.

Configuration Example

The following is an example configuration.

```
config
   instance instance-id 1
```

```
endpoint pfcp
   interface sxa
      overload-control msg-type session-report
      rate-limit 4500 queue-size 2500 reject-threshold 80 pending-request 2400
      exit
   profile sgw sgw1
   ddn delay-exclude-arplist [ 9 ]
   end
```

Configuration Verification

To verify the configuration:

```
#show running-config instance instance-id 1 endpoint pfcp
instance instance-id 1
endpoint pfcp
.
.
.
interface sxa
.
.
.
overload-control msg-type session-report
msg-priority high rate-limit 5 priority 1 queue-size 11 reject-threshold 80 pending-request
 10
exit
.
.
.
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the PFCP Session Report with DLDR Throttling Support feature.

Protocol Pod-level SXa Statistics

```
Query: sum(irate(proto_udp_req_msg_total{interface_type="SXA",
message_name=~"session_report_.*"}[1m])) by (message_name, status)
```

```
Legend: {{message_name}} | {{status}}
```

- **accepted**: Session reports accepted because of ARP configured in exclude-arp list
- **throttle_allow**: Session reports allowed by rate limit framework
- **throttled_pending_req_limit**: Session reports throttled by rate limit framework.

SGW Service-level Session Report Statistics

```
Query: sum(irate(sgw_sx_session_report_stats{}[30s])) by (sx_session_report_type,
 status)
```

```
Legend: {{sx_session_report_type}} -> {{status}}
```

Timesaver

For more information on bulk statistics support for SMF, see the *UCC 5G SMF Metrics Reference* document.

For more information on bulk statistics support for cnSGW-C, see the *UCC 5G cnSGW-C Metrics Reference* document.

Throttling Support for Create Session Requests on S11 Interface

Feature Description

In a live network deployment, due to high-traffic mobile networks, the S11 GTPC endpoint can receive a large number of Create Session Request messages, especially during peak times or in densely populated areas.

When the GTPC receives large number of CSReqs at GTPC pod on S11 interface, it tries to accept and process all requests. This can increase an overall load on the system. When the multiple CSReqs call occurs in a short time, there's a huge process overhead on the system. The Rate Limiting Configuration for S11 GTPC Endpoints enables GTPC to limit the number of CSReqs that enter the system to prevent the out of memory situation on the system.

How it Works

With Rate Limiting configuration, you can control the flow of incoming Create Session Request messages to the GTPC endpoints. This ensures that the GTPC endpoint can handle high traffic volumes effectively.

The GTPC pod uses the app-infra feature of adding virtual ID while creating new transaction to throttle the incoming messages on the basis of interface and request type.

The system takes appropriate action for the incoming messages that are based on the interface-level threshold configuration. The incoming messages get added to different queues and they are processed based on the configuration.

Once the configured threshold hits, CSReqs will start getting rejected. This configuration helps manage high traffic volumes by controlling the number of Create Session Requests that the GTPC endpoint can process at any given time.

Enable Throttling for Create Session Requests

With rate limiting configuration, you can manage the flow of incoming Create Session Request messages to the GTPC endpoints through the S11 interface. This ensures that the GTPC endpoint can efficiently handle high traffic volumes.

Important If the rate limiting configuration is enabled, it will also apply to emergency calls and WPS sessions, potentially causing Create Session Request messages for these calls to be rejected or dropped.

Notes:

- Both GRs instances must have the same rate limiter configuration.
- Create Session Request throttling must be configured in such a way that the waiting time in the queue should not exceed transaction SLA (pending-request/rate-limit < Txn SLA (in seconds)).
  - For converge core deployment without cnSGW, it is recommended to add a Create Session Request throttling configuration for the S5 interface only.
  - For converge core deployment with cnSGW, it is recommended to add a Create Session Request throttling configuration for both S11 and S5 interface at the same time.
- After adding this configuration, the gtpc-ep pod will be restarted.

Follow this procedure to enable rate limiting of Create Session Request messages at the GTPC endpoint.

Procedure

Step 1 Specify the instance ID.

instance instance-id *instance_id*

Example:

```
[smf] smf# config
[smf] smf(config)# instance instance-id 1
```

Step 2 Configure the GTP endpoint.

endpoint gtp

Step 3 Specify the interface.

interface s11

Step 4 Enable overload control for Create Session Request messages.

overload-control msg-type create-session-request

Step 5 Specify the action to reject the create session request messages when threshold reached for requests in virtual queue.

reject-action reject_req

Step 6 Set the rate limit for virtual queue and other parameters as desired.

rate-limit *rate_limit* **queue-size** *queue_size* **reject-threshold** *reject_threshold* **pending-request** *pending_requests*

Example:

```
[smf] smf(config-interface-s5)# rate-limit 3000 queue-size 8000 reject-threshold 100
pending-request 5000
```

Configuration Example

The following is an example configuration.

```
 config
   instance instance-id 1
      endpoint gtp
         interface s11
            overload-control create-session-request
            reject-action reject-req
            rate-limit 3 queue-size 10 reject-threshold 100 pending-request 2
            exit
        exit
    exit
exit
```

Configuration Verification

To verify the configuration, use the following show command:

```
#show running-config instance instance-id 1 endpoint gtp
instance instance-id 1
endpoint gtp
.
.
.
interface s11
.
.
.
overload-control msg-type create-session-request
reject-action reject-req
rate-limit 3 queue-size 10 reject-threshold 100 pending-request 2
exit
.
.
.
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the Create Session Requests Throttling feature.

- **rej_req_throttled_pending_req_limit**: Total number of the Create Session Request messages that were rejected when pending requests limit reached.
- **rej_req_throttled_queue_full**: Total number of the Create Session Request messages that were rejected when transaction queue is full.

The following is an example of the cnSGW service level metrics.

```
virtual_message_Pending_req_total{app_name="SGW",cluster="Local",data_c
enter="DC",instance_id="0",interface="s11",msg_type="createsessionrequest",
service_name="s11-gtpc-ep1",virtual_msg_id="1"} 0
virtual_message_dequeue_rate_total{app_name="SGW",cluster="Local",data_c
enter="DC",instance_id="0",interface="s11",msg_type="createsessionrequest",
service_name="s11-gtpc-ep1",virtual_msg_id="1"} 11
virtual_message_drop_total{app_name="SGW",cause="PendingRequestsLimitReached",cluster="Local",data_center="DC",instance_id="0",interface="s11",
msg_type="createsessionrequest",service_name="s11-gtpcep1",
virtual_msg_drop_code="8004",virtual_msg_id="1"} 19
<<<<<<<virtual_message_queued_total{app_name="SGW",cluster="Local",data_cent
er="DC",instance_id="0",interface="s11",msg_type="createsessionrequest",ser
vice_name="s11-gtpc-ep1",virtual_msg_id="1"} 11
<<<<<virtual_message_rate_limit_reached_total{app_name="SGW",cluster="Local",d
ata_center="DC",instance_id="0",interface="s11",msg_type="createsessionreq
uest",service_name="s11-gtpc-ep1",virtual_msg_id="1"} 6
```

Resiliency Handling

Feature Description

The Resiliency Handling feature introduces a CLI-controlled framework to support the service pod recovery, when you observe a system fault or a reported crash. It helps in recovering one of the following service pods:

- sgw-service pod
- smf-service pod
- gtpc-ep pod
- protocol pod

These service pods are software modules containing the logic to handle several session messages. The service pods are fault-prone due to any one of the following or a combination of multiple scenarios:

- Complex call flow and collision handling
- Inconsistent session state
- Incorrect processing of inbound messages against the session state
- Unexpected and unhandled content in the inbound messages

Whenever you observe the system fault or a crash, the fault behavior results into a forced restart of the service pod. It impacts the ongoing transaction processing of other sessions. The crash reoccurs even after the pod restart.

To mitigate this risk, use the CLI-based framework with actions defined to clean up subscriber sessions or terminate the current processing.

How it Works

This section describes how you can use the fault recovery framework to define actions for the crash. The framework allows you to define any of the following actions:

- Terminate—When a fault occurs, this action terminates the faulty transactions, and clears the subscriber session cache. It’s applicable for smf-service and sgw-service pods.

Note The pod doesn't get restarted. The database doesn't get cleared during this action.

- Cleanup—When a fault occurs, this action clears the faulty subscriber session and releases the call. It’s applicable for smf-service and sgw-service pods.
- Graceful reload—When a fault occurs, this action restarts the pod. It’s applicable for gtpc-ep, protocol, and pods. It handles the fault signals to clean up resources, such as the keepalive port and closes it early. It also allows the checkport script to detect the pod state and initiates the VIP switch processing for the corresponding pods.
- Reload—When the pod crashes, it initiates the reloading activity. It’s a default setting or value applicable for all the pods.

Feature Configuration

To configure this feature and to enable the system fault recovery, use the following sample configuration:

```
config
   system-diagnostics {   gtp | pfcp | service | sgw-service }
      fault
         action { abort | cleanup { file-detail | interval | num | skip
 { ims | emergency | wps } } | graceful-Reload | reload }
         end
```

NOTES:

- **system-diagnostics { gtp | pfcp | service | sgw-service }**—Specify the required type of service pods for system diagnostics. The available pod options are , gtp, pfcp, smf-service, and sgw-service.
- **fault**—Enables fault recovery while processing sessions.
- **action { abort | cleanup | graceful-Reload | reload }**—Specify one of the following actions to take on fault occurrence. The default action is reload.
  - **abort**—Deletes the faulty transaction and clears its session cache. The database doesn't get cleared.

Note It's an exclusive option to the smf-service pod.

  - **cleanup { file-detail | interval | num | skip }**—Enable the cleanup activity. It has the following selections to mitigate the fault action:
    - **file-detail**—Lists the file names with line numbers. It excludes the file name details from the recovery.
    - **interval**—Specifies the duration of the interval in minutes. This duration specifies the permissible interval within which it allows the maximum number of faults. Must be an integer in the range 1–3600.

- **num**—Specifies the maximum number of tolerable faults in an interval. Must be an integer in the range 0–50.
- **skip { ims | emergency | wps }**—Enable the skip cleanup of a subscriber session for an active voice call, or the WPS, or an emergency call.
  - To detect the active voice calls, use the following command:

    profile dnn *dnn_name* **ims mark qci** *qos_class_id*
  - When you enable the skip cleanup configuration, the 5GSA GW deletes the faulty transaction, and clears its session cache.
  - When a fault occurs during the session setup or the release state, the 5GSA GW performs the following:

    – Deletes the transactions on the session end.

    – Overrides the configured fault action during these states.

    – Clears the session cache and database entries for the faulty transaction.
  - It allows the dynamic configuration change.

Note It's an exclusive option to smf-service and sgw-service pods.

- **graceful-Reload**—Specify the option to gracefully reload the pod. The service pod handles fault signals to clean up resources like the keepalive port and continues with crash processing (pod restart processing).

Note It's an exclusive option to , gtpc-ep, and protocol service pods.

- **reload**—Reloads the pod, when it crashes due to a faulty behavior. It's an option applicable to all the service pods. It’s also the default option.

Configuration Example

The following example configuration allows three crashes of smf-service or sgw-service pods, within a duration of 10 minutes interval, and with the fault occurrence action as subscriber cleanup.

```
config
    system-diagnostics { service | sgw-service }
        fault
            num 3 interval 10
            action cleanup
            end
```

The following example configuration allows graceful fault handling for the or gtpc-ep pod or the protocol pod to close the keepalive port on receiving a fault signal.

Configuration Verification

To verify the configuration:

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following bulk statistics are supported for the resiliency handling feature.

recover_request_total—This statistic includes the following new labels:

- **action**—Defines the fault action.
- **reason**—Defines the fault reason.
- **status**—Defines the fault status.

The following is an example of bulk statistics for the resiliency handling feature.

```
recover_request_total{action="panic_recovery_cleanup",
app_name="SMF",cluster="Local",data_center="DC",instance_id="0",
reason="creating panic",service_name="sgw-service",status="success"} 1
```

For more information on bulk statistics support for 5GSA GW, see the *UCC 5G 5GSA GW Metrics Reference* document.

For more information on bulk statistics support for cnSGW-C, see the *UCC 5G cnSGW-C Metrics Reference* document.

Roaming Peer Path Management Optimization

Feature Description

cnSGW-C supports inbound roaming. When inbound roaming is enabled, cnSGW-C communicates with remote PGW which is located in the roamer home network.

cnSGW-C generates Echo Request messages towards the roaming peers and detects path failure, thereby handling echo request messages from the roaming peers.

How it Works

This section describes how this feature works.

cnSGW-C uses subscriber policy and operator policy to categorize peer as a roamer or a home peer. cnSGW-C applies the following functionalities to the roaming peer:

- cnSGW-C responds immediately to the echo request message received from the roaming peer. If the restart counter value changes in the echo request, cnSGW-C doesn't detect path failure towards the peer.

- cnSGW-C continues to generate echo request towards the roaming peer after reaching the configured echo interval. If the restart counter value changes in the echo response, cnSGW-C detects path failure towards the peer.
- If the restart counter value changes in the first Create Session Response message and the SGW Relocation Modify Bearer Response message, cnSGW-C detects path failure towards the peer.
- cnSGW-C doesn't update the last activity time of roaming when it receives an echo request from the roaming peer.
- In earlier implementation, Nodemgr environment variable ROAMING_PEER_ECHO_MODULATOR controls the echo request generation towards the roaming peers. Modifying the value of this environment variable causes the NodeMgr pod to restart every time after the change.

  To optimize this behaviour, use of environment variable ROAMING_PEER_ECHO_MODULATOR is deprecated and support is added to configure different echo interval for roaming peers using S8e interface configuration. If the S8e interface echo configuration is enabled, then cnSGW-C uses this configuration for Echo Request generation towards roaming peer.

  For more information, refer to Enable the echo request optimization for roaming peer node.
- If S8e interface configuration is not present then cnSGW-c uses the Environment variable ROAMING_PEER_ECHO_MODULATOR to control echo request generation towards the roaming peer. The default value for ROAMING_PEER_ECHO_MODULATOR is 3. cnSGW-C generates echo request towards the roaming peer after reaching ROAMING_PEER_ECHO_MODULATOR * Echo Interval.

  For example, if the ROAMING_PEER_ECHO_MODULATOR is 3, and the Echo Interval is 60, the cnSGW-C generates the Echo Request after 180 seconds. Similarly, if the ROAMING_PEER_ECHO_MODULATOR is 0, cnSGW-C doesn't generate the echo request towards the roaming peer.
- In the GTPC-EP pod, the variable GTPC_UPDATE_LAST_MSG_RECV_TIME_AFTER controls the last activity time updates. The default value for this variable is 30 seconds. The cnSGW-C updates the last activity time for the peer after 30 seconds. You can increase this value to reduce the last activity time update notifications towards the NodeMgr.

Feature Configuration

Configuring this feature involves the following steps:

- Configure the operator policy with the roaming status as roamer, and associate the operator policy with the subscriber policy to identify the operator as roaming peer. For more information, see Configuring the Operator Policy and Subscriber Policy, on page 507.
- Configure the default gateway to be used, while adding the BGP route dynamically. For more information, see Configuring the Default Gateway, on page 509.
- Enable the echo request optimization for roaming peer node, see Configuring the echo interval for Roaming Peer Node

Configuring the Operator Policy and Subscriber Policy

To configure this feature, use the following configuration:

```
config
  policy
    subscriber subscriber_name
      precedence precedence_value
        imsi mcc mcc_value
        imsi mnc mnc_value
        operator-policy policy_name
        exit
      exit
    operator operator_policy
      roaming-status roamer
      exit
  profile sgw sgw_profile_name
      subscriber-policy policy_name
      end
```

NOTES:

- **precedence** *precedence_value*—Specify the precedence for entry. Must be an integer in the range of 1-2048.
- **mcc** *mcc_value*—Specify the Mobile Country Code (MCC). Must be a three-digit number.
- **mnc** *mnc_value*—Specify the Mobile Network code (MNC). Must be a two or three-digit number.
- **operator-policy** *policy_name*—Specify the operator policy name. Must be a string.
- **policy-operator** *operator_policy*—Specify the operator policy. Must be one of the following:
  - <any string>
  - defOprPoll
  - opPolHomer
  - opPolRoaming
  - opPolVisiting
  - opPolVisiting_hrt
  - opPolVisiting_hrt_overriden
- **roaming-status roamer**—Specify the roaming status of the peer. This is disabled by default.
- **subscriber-policy** *policy_name*—Specify the subscriber policy name. Must be a string.

Configuration Example

The following is an example configuration.

```
config
   policy subscriber polSubSgw
      precedence 1
         imsi mcc 310
         imsi mnc 260
         operator-policy Home_op1
         exit
```

```
   precedence 2
      imsi mcc 311
      imsi mnc 660
      operator-policy Home_op1
      exit

   precedence 3
      imsi mcc 310
      imsi mnc 240
      operator-policy Home_op1
      exit

   precedence 4
      operator-policy Roaming_SGW_op1
      exit
   exit

   policy operator Roaming_SGW_op1
      roaming-status roamer
      exit

   profile sgw sgw1
      subscriber-policy polSubSgw
      end
```

Configuring the Default Gateway

To configure this feature, use the following configuration:

```
config
   router bgp local_as_number
   policy-name policy_name
      source-prefix source_prefix_value
      mask-range mask_range
      interface interface_id
      gateWay gateway_address
      end
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **policy-name** *policy_name*—Specify the policy name.
- **source_prefix** *source_prefix_value*—Specify the source prefix value.
- **mask-range***mask_range*—Specify the mask range.
- **gateWay** *gateway_address*—Specify the gateway address.

Configuration Example

The following is an example configuration.

```
config
   router bgp 6500
   policy-name sgw_bgp
      source-prefix 209.165.201.12/32
      mask-range 32..32
      interface ens224.2084
```

```
gateWay 209.165.201.28
end
```

Configuration Verification

To verify the configuration:

- **show subscriber namespace sgw imsi 123456789012345 full subscriber-details**

```
{
"subResponses": [
  "pdnInfoList": {
     "pdnInfo": [
          {
           "plmnType": "VISITOR"
           "s5ePeerType": "ROAMER"
           }
      ]
   }
]
}
```

- **show peers | include S5**

```
1 S5E 209.165.201.12:212320.20.20.124:2123Inbound nodemgr-1 Udp 2 minutes PGW
MaxRemoteRcChange: N/A,Recovery: 10 S5
1 S5E 209.165.201.12:212320.20.20.127:2123Inbound nodemgr-0 Udp 35 seconds PGW
MaxRemoteRcChange: N/A,Recovery: 10,S5E PeerType: Roaming
```

Enable the echo request optimization for roaming peer node

Follow these steps to enable the echo request optimization for roaming peer node.

Procedure

Step 1 Configure echo interval on S8e interface.

Step 2 Configure the path failure detection policy.

Step 3 Associate path failure detection policy on S8e interface, on page 512.

Configure echo interval on S8e interface

To enable the echo request on the S8e interface, use the following steps:

Procedure

Step 1 Enter the Instance ID Configuration mode.

instance instance-id *instance_id*

Example:

```
[smf] smf# config
[smf] smf(config)# instance instance-id 1
```

Step 2 Configure the GTPC endpoint.

endpoint *endpoint_type*

Example:

```
[smf] smf(config-instance-1)# endpoint gtp
```

Step 3 Configure the S8e interface.

interface *interface_type*

Example:

```
[smf] smf(config-endpoint-gtp)# interface s8e
```

Step 4 Configure the echo.

echo interval *echo_interval* **retransmission-timeout** *retransmission_timeout* **max-retransmissions** *max_retransmissions*

Example:

```
[smf] smf(config-interface-s8e)# echo interval 300 retransmission-timeout 10
max-retransmissions 4
```

Note

- *echo_interval*—Specify the echo interval in seconds. Must be an integer in the range of 60 to 3600. Default value is 300. Use 0 to disable the echo request generation to the roaming peer.
- *retransmission_timeout*—Specify the echo retransmission timeout in seconds. Must be an integer in the range of 1 to 20. Default value is 5.
- *max_retransmissions* —Specify maximum retries for GTP echo request. Must be an integer in the range of 0 to 15. Default value is 3.

Step 5 Save and commit the configuration.

Example:

```
[smf] smf(config-interface-s8e)# end
```

Step 6 [*Optional*] Use **show running-config instance instance-id 1 endpoint gtp interface s8e** command to verify if the echo interval is configured.

Example:

```
[sgw] smf# show running-config instance instance-id 1 endpoint gtp interface s8e
Thu Dec  12 09:26:46.295 UTC+00:00
instance instance-id 1
 endpoint gtp
  interface s8e
   echo retransmission-timeout 10
   echo max-retransmissions 4
   echo interval          300

  exit
 exit
exit
```

Associate path failure detection policy on S8e interface

To enable the echo request on S8e interface, use the following steps:

Procedure

Step 1 Enter the Instance ID Configuration mode.

instance instance-id *instance_id*

Example:

```
[smf] smf# config
[smf] smf(config)# instance instance-id 1
```

Step 2 Configure the gtp endpoint.

endpoint *endpoint_type*

Example:

```
[smf] smf(config-instance-1)# endpoint gtp
```

Step 3 Configure the s8e interface.

interface *interface_type*

Example:

```
[smf] smf(config-endpoint-gtp)# interface s8e
```

Step 4 Define the path failure detection policy.

path-failure detection-policy *detection_policy_name*

Example:

```
[smf] smf(config-interface-s8e)# path-failure detection-policy pf1
```

Note *detection_policy_name*—Specify the failure detection policy name. Must be a string.

Step 5 Save and commit the configuration.

Example:

```
[smf] smf(config-interface-s8e)# end
```

Step 6 [*Optional*] Use **show running-config instance instance-id 1 endpoint gtp interface s8e** command to verify if the path failure detection policy is enabled.

Example:

```
[sgw] smf# show running-config instance instance-id 1 endpoint gtp interface s8e
Thu Dec  12 09:26:46.295 UTC+00:00
instance instance-id 1
 endpoint gtp
  interface s8e
   echo retransmission-timeout 10
```

```
   echo max-retransmissions 4
   echo interval          300
   path-failure detection-policy pf1
  exit
 exit
exit
```

Configure the path failure detection policy

To configure the path failure detection policy, use the following steps:

Procedure

Step 1 Enter the path failure detection configuration mode.

path-failure detection-policy*detection_policy_name*

Example:

```
[smf] smf# config
[smf] smf(config)# path-failure detection-policy pf1
```

Step 2 Configure the path failure detection policy with maximum remote restart counter change and ignore.

max-remote-rc-change *max_remote_rc_change* **ignore** *ignore*

Note *ignore*—Specify to ignore restart counter values, echo timeouts, or echo failures. Must be **control-rc-change**, **echo-failure**, or **echo-rc-change**.

Example:

```
[smf] smf(config-path-failure-detection-pf1)# max-remote-rc-change  2 ignore echo-failure
```

Step 3 Save and commit the configuration.

Example:

```
[smf] smf(config-path-failure-detection-pf1)# end
```

Step 4 [*Optional*] Use **show running-config policy path-failure-detection** command to verify if the path failure detection policy is configured.

Example:

```
[sgw] smf# show running-config policy path-failure-detection
Thu Dec  12 09:26:46.295 UTC+00:00
Thu Dec  12 09:27:17.678 UTC+00:00
policy path-failure-detection pf1
 max-remote-rc-change 2
 ignore echo-failure
exit
```

Sample configuration for optimizing the echo for roaming peer node

This section provides the sample configuration to enable echo optimization on the S8e interface:

```
config
instance instance-id 1
 endpoint gtp
  interface s8e
   echo retransmission-timeout 10
   echo max-retransmissions 4
   echo interval          300
   path-failure detection-policy pf1
  exit
 exit
exit

config
policy path-failure-detection pf1
 max-remote-rc-change 2
 ignore echo-failure
exit
```

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the Roaming Peer Path Management Optimization feature.

gtpc-ep statistics indicating Echo Request Handling from Roaming Peer is Suppressed:

```
gtpc_roaming_peer_path_mgmt{app_name="SGW",cluster="Local",data_center="DC",
gtpc_peer_type="ROAMER",instance_id="1",interface_type="S5E",service_name="gtpc-ep",
status="suppressed"} 1
```

upd_proxy statistics indicating Total number of bgp add request:

```
# HELP upd_proxy_bgp_routes_count Total number of bgp add request
# TYPE upd_proxy_bgp_routes_count counter
upd_proxy_bgp_routes_count{app_name="SGW",cluster="Local",data_center="DC",
gr_instance_id="1",instance_id="0",service_name="udp-proxy",status="success"} 1
```

Flag DB Database Updates

Feature Description

cnSGW-C updates the CDL whenever the subscriber state changes from idle to active, and the ULI, UeTz, UCI, or the serving network is modified.

When the transaction requests driven to CDL increases, cnSGW-C incurs a higher CPU utilization. To prevent the needless CPU utilization, cnSGW-C updates only a subset of the CDL with the changed attributes.

Flag DB Database for the DDN Procedure

When the DDN procedure completes, sgw-service updates the CDL which impacts the CPU utilization. To optimize the CPU usage, the CDL is notified about the DDN only with the partial updates.

DDN Internal Timer

cnSGW-C implements the DDN Retry Timer by applying the CDL's timer functionality. Every DDN transaction starts the DDN Retry Timer that requires the complete CDL instance to be updated, which results in an increase in the CPU usage of the CDL and sgw-service.

cnSGW-C is modified to have an integrated DDN Retry Timer that is configurable from sgw-profile. With this approach, the performance is improved because the cnSGW-C does not communicate with the CDL for starting the DDN Retry Timer as it is an internal timer. The DDN Retry Timer is started for a duration of 10 seconds.

OAM Support

This section describes operations, administration, and maintenance support for this feature.

Bulk Statistics Support

The following statistics are supported for the Flag DB Database Updates feature:

- mbr_partial_cdl_update: Captures the total number of partial CDL update procedures invoked by the Modify Bearer Request.

  Sample query:

```
sgw_cdl_update_stats{app_name="smf",cdl_update_type="
mbr_partial_cdl_update",cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",rat_type="EUTRAN",service_name="sgw-service"} 1
```

- ddn_partial_cdl_update: Captures the total number of partial CDL update procedures that DDN has invoked

  Sample query:

```
sgw_cdl_update_stats{app_name="smf",cdl_update_type="
ddn_partial_cdl_update",cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",rat_type="EUTRAN",service_name="sgw-service"} 1
```

For more information on bulk statistics support, see *UCC Serving Gateway Control Plane Function Metrics Reference*.

UDP Proxy Functionality Merged into Protocol Microservices

Feature Description

The UDP proxy microservices provide UDP transport termination for protocols (PFCP, GTPC, and RADIUS) that require UDP protocol as the transport layer protocol. The UDP proxy provides user space packet forwarding and IPC communication to protocol microservices. It uses host networking for source IP address observability and operates in Active-Standby mode.

Multiple protocol microservices depend on UDP proxy for UDP transport. Therefore, UDP proxy is a scaling bottleneck. A surge of messages can lead to packet drop.

The incoming and outgoing messages use the UDP proxy pod for forwarding messages. With minimal packet processing, the UDP proxy forwards the messages to the GTPC-EP pod. This requires the IPC communication for message forwarding, along with marshal or unmarshal of the packet.

The UDP proxy functionality merges into the respective protocol microservice to mitigate the scaling bottleneck. The protocol pod receives the messages directly, and avoids forwarding the messages and IPC communication.

The UDP proxy bypass improves the CPU usage by reducing one hop across microservices in the signaling path. cnSGW-C supports UDP proxy bypass for the PFCP and GTPC protocols.

PFCP Protocol Endpoint with UDP Proxy Bypass

With the UDP proxy mode, all message exchanges for the protocols, such as N4, Sxa, and GTP-U, occur through the UDP proxy. The UDP proxy is responsible for connecting or receiving connections from the UPF. The service or the UPF initiates all node-related or session-related messages, and the responses pass through the UDP proxy. The UDP proxy handles all node-related messages and forwards the messages to the protocol node.

With the outbound UDP proxy bypass mode, the session-related messages flow directly from the protocol to the UPF. The node-related messages continue to take the current path, which is through the UDP proxy to the protocol pod or the node manager.

With the inbound and outbound UDP proxy bypass mode, the service sends the session-related messages directly to UPF through the protocol pod, with UDP proxy bypassed. The protocol also establishes a connection with the UPF as and when the app service initiates a PFCP message toward the UPF.

For more information about UDP Proxy Bypass for PFCP, see *UCC 5G SMF Configuration and Administration Guide*.

GTPC Protocol Endpoint with UDP Proxy Bypass

With the UDP proxy mode, all message exchanges for the GTPv2 protocol occur through the UDP proxy. The UDP proxy is responsible for connecting or receiving connections on the S11 and S5e, S2b, and S5/S8 interface. The service or the GTP peer initiates the session-related or node-related messages, and the responses pass through the UDP proxy. The UDP proxy handles all node-related messages and forwards the messages to the protocol node.

With the inbound and outbound UDP proxy bypass mode, the service-initiated session-related messages are sent directly to the GTP peer through the GTPC-EP pod, with the UDP proxy bypassed. For node-related messages, the GTPC-EP starts a GTP endpoint for peers to connect with it on the S11, S5e, S2b, and S5/S8

interfaces. The GTPC-EP pod also establishes a connection with the GTP peer as and when the app service initiates a GTPv2 message toward the GTP peer.

The following features are integrated from UDP proxy:

- Transaction SLA
- DSCP marking for GTP packets
- Adding BGP routes for roamer subscribers on the fly
- Supporting Dispatcher feature and incoming retransmission
- SGW cache integration for DDN
- MBR cache integration

The following features are integrated from GTPC-EP:

- Retransmissions based on n3t3 configuration for outbound requests
- Monitor protocol and monitor subscriber
- Echo message handling

All existing features supporting the UDP proxy mode are supported with and without the UDP proxy bypass mode.

How it Works

This section describes how this feature works.

The GTPC-EP k8 service is disabled when the bypass feature is enabled.

The GTPC protocol endpoint with the UDP Proxy Bypass feature requires the GTPC-EP pod to run in the host environment in Active-Standby mode. When the GTPC-EP pod runs in the host environment in the Active-Standby mode, the k8-service is disabled. Further, if the pods (SGW-Service and the node manager) must communicate with the GTPC-EP pod, an extra endpoint is required at the GTPC-EP pod. This infra endpoint initializes at the GTPC-EP app start and the internal IP is used for the same.

When the internal IP is not configured, the available GTP VIPs are used for initializing the infra endpoint.

The S11, S5, S5e, and S2b interfaces are used to configure the GTP VIPs instead of the base GTP VIP IP address.

The UDP sockets are created at the GTPC-EP pod for handling GTP packets.

No new CLI or keyword is added to enable or disable bypass UDP proxy functionality. The existing endpoint configuration is used in the following manner to enable or disable bypass UDP proxy functionality:

- GTP VIPs must be configured under the endpoint protocol for using UDP proxy (no bypass).
- GTP VIPs must be configured under the GTPC endpoint to enable bypass UDP proxy.
- If the GTP VIPs are configured under both the protocol endpoint and the GTP endpoint, the UDP proxy is used by default.
- The GTPC feature-specific configurations, such as Retransmission n3t3-based, ECHO, SLA, Dispatcher, and the DSCP must be configured under endpoint GTP irrespective of the bypass feature.

Note Prior to this feature, if endpoint GTP was configured, the UDP proxy mode was the default behaviour. With this feature onwards, if the endpoint GTP is configured with the GTP interface VIP (s5, s11, s5e, or s2b), the UDP proxy bypass is enabled by default. For UDP proxy bypass to be disabled, the endpoint protocol must be configured with the GTP interface VIP (s5, s11, s5e, or s2b).

Call flows

This section describes the key call flows for this feature.

GTPC Protocol Endpoint with UDP Proxy Bypass Call Flow

This section describes the GTPC Protocol Endpoint with UDP Proxy Bypass call flow.

Figure 108: GTPC Protocol Endpoint with UDP Proxy Bypass Call Flow

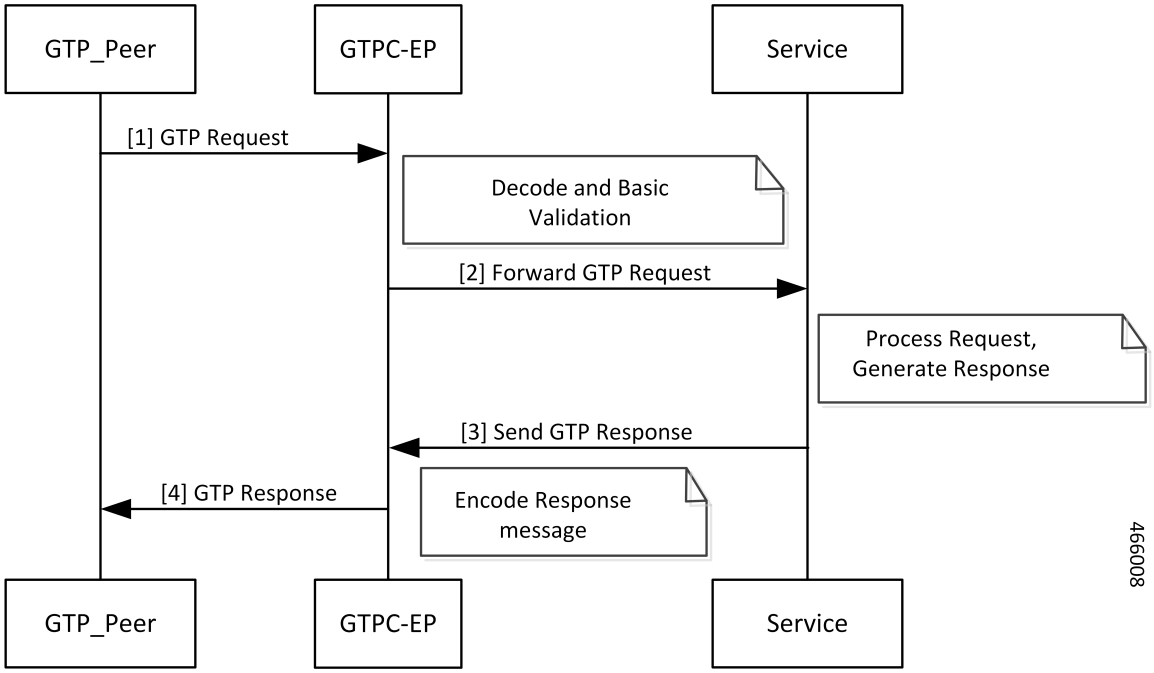


Table 190: GTPC Protocol Endpoint with UDP Proxy Bypass Call Flow Description

Step	Description
1	The GTP_Peer sends the GTP Request to the GTPC-EP pod.
2	The GTPC-EP pod decodes the message, performs the basic validation, and forwards the GTP Request to the Service.
3	The Service processes the GTP Request, generates the GTP Response, and sends the GTP Response to the GTPC-EP pod.
4	The GTPC-EP pod encodes the GTP Response message and forwards the GTP Response message to the GTP_Peer.

Feature Configuration

Configuring this feature involves the following steps:

- Configuring the internal VIP
- Configuring the GTP VIPs under the interface for initializing the infra-GTP endpoint

Configuring Internal VIP

To configure the internal VIP, use the following configuration:

```
config
  instance instance-id instance_id
    endpoint gtp
      internal-vip  vip_ip_address
      vip-ip ipv4_address vip-port ipv4_port_number
      vip-ipv6 ipv6_address vip-ipv6-port ipv6_port_number
      dual-stack-transport { true | false }
      end
```

NOTES:

- **internal-vip** *vip_ip_address*—Specify the internal VIP IP address of the additional endpoint of the GTPC-EP pod. The Service pod sends the messages directly to this IP address.
- **dual-stack-transport { true | false }**—Enable the dual stack feature that allows you to specify IPv6 or IPv4 address. Specify true to enable this feature.

Configuring GTP VIPs

To configure the GTP VIPs under the interface for initializing the infra-GTP endpoint, use the following configuration:

```
config
  instance instance-id instance_id
    endpoint gtp
      vip-ip ipv4_address vip-port ipv4_port_number
      vip-ipv6 ipv6_address vip-ipv6-port ipv6_port_number
      dual-stack-transport { true | false }
    interface interface_name
      vip-ip ipv4_address vip-port ipv4_port_number
      vip-ipv6 ipv6_address vip-ipv6-port ipv6_port_number
      dual-stack-transport { true | false }
      end
```

NOTES:

- **vip-ip** *ipv4_address* **vip-port***ipv4_port_number*—Specify the IPv4 address of the interface.
- **vip-ipv6** *ipv6_address* **vip-ipv6-port***ipv6_port_number*—Specify the IPv6 address of the interface.
- **dual-stack-transport { true | false }**—Enable the dual stack feature that allows you to specify IPv6 or IPv4 address. Specify true to enable this feature.

Configuration Example

The following are example configurations.

Example configuration with the internal VIP for GTPC-EP, with UDP Proxy bypass enabled:

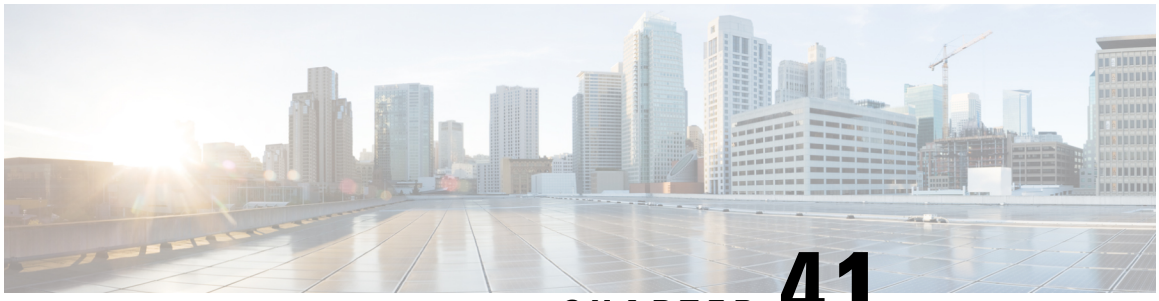
```
config
    instance instance-id 1
        endpoint gtp
          internal-vip 209.165.201.15
          end
```

Example configuration with the bypass feature enabled (UDP Proxy bypassed):

```
config
    instance instance-id 1
        endpoint gtp
          vip-ip 209.165.201.20
          interface s5
            vip-ip 209.165.201.20
            exit
          interface s5e
            vip-ip 209.165.201.8
            exit
          interface s2b
            vip-ip 209.165.201.20
            exit
          interface s11
            vip-ip 209.165.201.8
            end
```

Example configuration with bypass feature disabled (UDP Proxy used for GTP messages):

```
config
    instance instance-id 1
        endpoint protocol
          vip-ip 209.165.201.20
          interface s5
            vip-ip 209.165.201.20
            exit
          interface s5e
            vip-ip 209.165.201.8
            exit
          interface s2b
            vip-ip 209.165.201.20
            exit
          interface s11
            vip-ip 209.165.201.8
            end
```



CHAPTER 41

Presence Reporting Area

- Feature Summary and Revision History, on page 521
- Feature Description, on page 521
- How it Works, on page 522

Feature Summary and Revision History

Summary Data

Table 191: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 192: Revision History

Revision Details	Release
First introduced.	2021.02.1

Feature Description

Presence Reporting Area (PRA) is an area defined within 3GPP packet domain for reporting UE presence within that area for policy control and/or charging reasons.

A Presence Reporting Area consists of:

- TAs or eNBs and/or ECGI for E-UTRAN
- RAs or SAIs or CGIs for UTRAN
- RAs or CGIs for GERAN

The two types of Presence Reporting Areas that apply to an MME pool are UE-dedicated Presence Reporting Areas and Core Network pre-configured Presence Reporting Areas.

Note cnSGW-C supports Core Network pre-configured PRAs in this release.

The information for leveraging PRA for implementing differential charging and policy enforcement is provided in *3GPP TS 23.401*.

The cnSGW supports:

- Passing PRA action to MME as received from PGW in CSRsp, CBReq, UBreq, MBRsp, and CNRsp messages.
- Passing PRA information to PGW if received from MME in CSReq, CBRsp, UBRsp, CNotf Req, and MBReq messages.
- Always passing the message to PGW if PRA information is present in MBReq.

How it Works

This section describes how this feature works.

cnSGW-C relays the instructions received from PGW towards MME and the information thus reported by MME towards PGW.

To pass the information, two IEs are defined on GTPC interface as per *3GPP TS 29274 V15.4.0*. The following section shows the IE support implemented in cnSGW-C.

Presence Reporting Area Action

Table 193: Presence Reporting Area Action

Octets	Bits							
	8	7	6	5	4	3	2	1
1	Type = 177							
2–3	Length = n							
4	SPARE				Instance			
5	SPARE				INAPRA	Action		
6–8	Presence Reporting Area Identifier							

Table 194: Action Values

Action	Value (Decimal)
Start Reporting changes of UE presence in the Presence Reporting Area (PRA)	1
Stop Reporting changes of UE presence in the Presence Reporting Area (PRA)	2

- The Action value 1 (Start Reporting change) is used to request to start reporting changes of UE presence in the Presence Reporting Area identified by the Presence Reporting Area Identifier and, if present, the Presence Reporting Area elements composing the Presence Reporting Area.
- The Action value 2 (Stop Reporting change) is used to request to stop reporting changes of UE presence in a Presence Reporting Area.
- The Inactive PRA (INAPRA) flag in the Octet 5 is set to 1 on the S10/S16/S3 interface if the PRA is inactive in the source MME/SGSN during an inter MME/SGSN mobility procedure, i.e. the reporting change of UE presence in this Presence Reporting Area was requested by the PGW/PCRF/OCS but it was deactivated by the source MME/SGSN.
- The Presence Reporting Area Identifier shall be present if the Action value requests to start, stop, or modify reporting changes of UE presence in a Presence Reporting Area. If so, the Presence Reporting Area Identifier shall contain an identifier of the Presence Reporting Area and encoded using full hexadecimal representation.

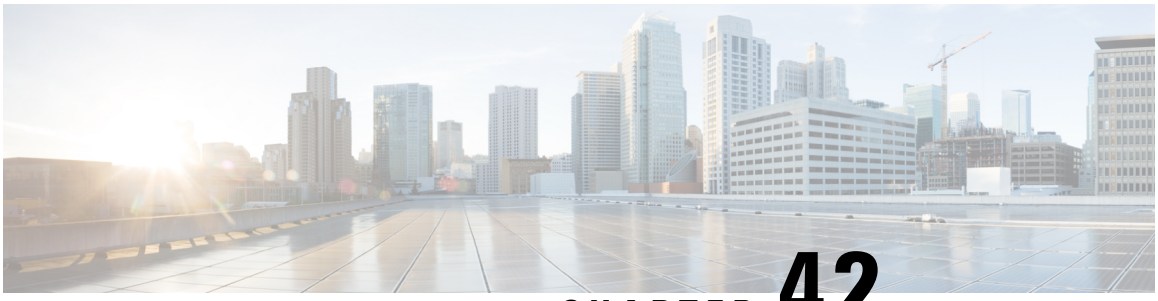
Presence Reporting Area Information

Table 195: Presence Reporting Area Information

Octets	Bits							
	8	7	6	5	4	3	2	1
1	Type = 178							
2–3	Length = n							
4	SPARE				Instance			
5–7	PRA Identifier							
8	SPARE				INAPRA	SPARE	OPRA	IPRA

- The PRA Identifier in octets 5–7 is present and contains the identifier of the PRA the UE is entering or leaving. It's encoded using full hexadecimal representation (binary, not ASCII encoding). The PRA Identifier is defined in *3GPP TS 23.003 [2], Clause 19.10*.
- The Inside PRA (IPRA) flag is set to 1 if the UE is inside or enters the Presence Reporting Area identified by the PRA Identifier.
- The Outside PRA (OPRA) flag is set to 1 if the UE is outside or leaves the Presence Reporting Area identified by the PRA Identifier.

- The Inactive PRA (INAPRA) flag in octet 8 is set to 1 if the PRA is inactive in the MME/SGSN, i.e. the reporting of change of UE presence in this PRA is currently deactivated in the MME/SGSN, e.g. due to an overload situation.
- Either the IPRA or the OPRA flag or the INAPRA is set to 1, not several ones, for a given Presence Reporting Area Identifier.



CHAPTER 42

NB-IoT and LTE-M RAT Type Support

Table 196: Feature History

Feature Name	Release Information	Description
RAT Type NB-IoT and LTE-M support	2025.03.0	cnSGW-C supports "RAT Type NB-IoT and LTE-M" to enable accurate identification and differentiation of Narrowband IoT (NB-IoT) and LTE-M devices in Charging Data Records (CDRs) for precise billing. Command introduced: **gtpp-attributes additional-rat-type [lte-m nb-iot]**— Used to enable RAT Type NB-IoT and LTE-M. **Default Setting**: Disabled – Configuration Required

RAT Type NB-IoT and LTE-M support refers to the ability of a network to recognize and differentiate between two specific types of Radio Access Technologies (RATs): **Narrowband IoT (NB-IoT)** and **LTE-Machine Type Communications (LTE-M)**. These are specialized technologies designed for **Internet of Things (IoT)** devices, offering optimized connectivity for low-power, wide-area (LPWA) applications.

Why Differentiation in RAT Type is Important

Both technologies coexist on the 4G or 5G core network but serve different purposes and use cases. Differentiating these RAT Types in network elements like Serving Gateways (SGWs) helps operators:

- Generate accurate Charging Data Records (CDRs) for billing.
- Enable specific configurations for optimized performance.
- Analyze metrics and monitor IoT device traffic based on their RAT Type.

- Enable RAT Type NB-IoT and LTE-M for CDR, on page 526
- Troubleshooting Information for RAT-Type Support, on page 526

Enable RAT Type NB-IoT and LTE-M for CDR

This task allows operators to configure RAT Type Information Elements (IE) for NB-IoT and LTE-M devices, ensuring precise billing, streamlined monitoring, and efficient network management.

Procedure

Step 1 Enter the SGW charging profile configuration mode.

profile sgw-charging-profile *profile_name*

Example:

```
[sgw] smf(config)# profile sgw-charging-profile ch1
```

Step 2 Enable RAT Type NB-IoT and LTE-M.

gtpp-attributes additional-rat-type [lte-m nb-iot]

Note If additional-rat-type is not configured in sgw-charging-profile, CDRs generated for LTE-M/NB-IOT devices have RAT Type IE as EUTRAN only.

Step 3 Save and commit the configuration.

Troubleshooting Information for RAT-Type Support

This section provides output of show commands, clear commands, and bulk statistics that can be used to troubleshoot the issues.

Show Commands

- **Show running config output:** To view additional-rat-type information, use the **show full-configuration** command.

```
   [sgw] smf(config)# show full-configuration profile sgw-charging-profile ch1
profile sgw-charging-profile ch1
 gtpp-attributes additional-rat-type [lte-m nb-iot]
exit
```

- **Show subscriber config output:** To view additional-rat-type subscriber information, use the **show subscriber** command.

```
[sgw] smf# show subscriber nf-service sgw rat lte-m
subscriber-details
{
  "subResponses": [
    [
      "id-index:1:0:32768",
      "id-index-key:1:0:globalKey:32768",
```

```
      "id-value:16778240",
      "imsi:imsi-123456789012348",
      "msisdn:msisdn-223310101010101",
      "imei:imei-123456786666660",
      "rat:lte-m",
      "upf:10.1.44.42",
      "upfEpKey:10.1.44.42:10.1.45.167",
      "subscribertype:non-volte",
      "s5s8Ipv4:10.1.43.219",
      "s11Ipv4:10.1.44.42",
      "data-tunnel:IPV4",
      "namespace:sgw",
      "nf-service:sgw"
    ]
  ]
}

[sgw] smf# show subscriber nf-service sgw count rat nb-iot
Tue Apr  29 08:36:31.184 UTC+00:00
subscriber-details
{
  "sessionCount": 1
}
```

Clear Commands

To clear the the additional-rat-type information for a subscriber, use the **clear subscriber** command.

```
[sgw] smf# clear subscriber nf-service sgw rat lte-m
```

Example of message notification post running the **clear subscriber** command.

Example - Clear subscriber request submitted successfully for GR Instance ID 1. Waiting Time is 9.540360 seconds.

Bulk Statistics Output

This feature introduces the following changes to the statistics related to the SGW service:

- New Label: `rat_type`
  - The `rat_type` label has been added to the following statistics:
    - `sgw_service_stats`
    - `sgw_ue_disconnect_stats`
  - This label indicates the Radio Access Technology (RAT) type used for the connection.
- Modified `rat_type` values
  - The implementation has been modified to populate the `rat_type` label with values of `nb-iot` or `lte-m` in the following statistics:
    - `sgw_ue_stats`
    - `sgw_pdn_stats`
    - `sgw_pdn_disconnect_stats`

Here are examples of the updated statistics with the new `rat_type` label and the updated values:

- sgw_service_stats

```
sgw_service_stats{app_name="smf",cluster="Local",
data_center="DC",fail_reason="",gr_instance_id="1",instance_id="0",
interface="interface_sgw_egress",rat_type="LTE-M",reject_cause="",
service_name="sgw-service",sgw_procedure_type="initial_attach",
status="attempted",sub_fail_reason="",svc_to_svc="False"} 1
```

- sgw_ue_stats

```
sgw_ue_stats{app_name="smf",cluster="Local",
data_center="DC",gr_instance_id="1",instance_id="0",
rat_type="LTE-M",service_name="sgw-service",status="setup"} 1
```

- sgw_pdn_stats

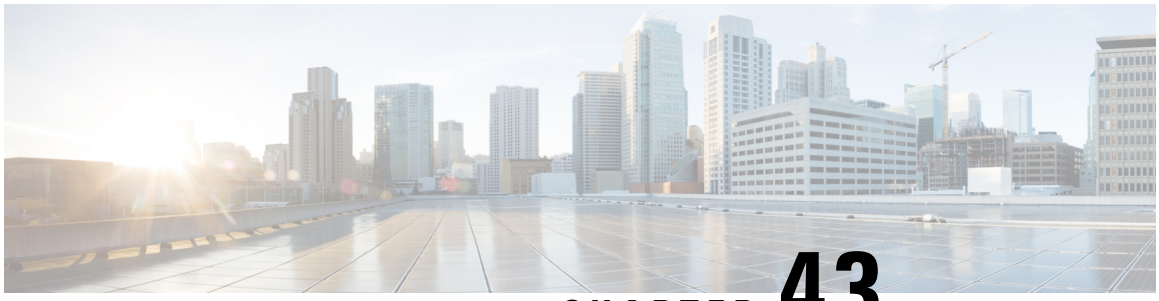
```
sgw_pdn_stats{app_name="smf",cluster="Local",
data_center="DC",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4",rat_type="LTE-M",
service_name="sgw-service",status="setup"} 1
```

- sgw_ue_disconnect_stats

```
sgw_ue_disconnect_stats{app_name="smf",cluster="Local",
data_center="DC",gr_instance_id="1",instance_id="0",peer_ip="",
rat_type="LTE-M",reason="remote_init_disconnect",service_name="
sgw-service"} 1
```

- sgw_pdn_disconnect_stats

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="Local",
data_center="DC",gr_instance_id="1",instance_id="0",pdn_type="ipv4",
rat_type="LTE-M",reason="mme_init_pdn_sess_rel",service_name=
"sgw-service"} 1
```



CHAPTER 43

Redundancy Support

- Feature Summary and Revision History, on page 529
- High Availability Support, on page 530
- Inter-Rack Redundancy Support, on page 533

Feature Summary and Revision History

Summary Data

Table 197: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Configuration Required
Related Documentation	Not Applicable

Revision History

Table 198: Revision History

Revision Details	Release
First introduced.	2021.02.0

High Availability Support

Feature Description

The cnSGW-C is built on the Kubernetes cluster strategy so that it inherits the high availability aspects of K8 cluster deployments. The cnSGW-C uses the construct that includes the components, such as pods and services.

Each pod has at least 2 instances to ensure high availability against:

- Pod instance restart or failure
- Pod lost due to node restart or failure

For details on the pods and services, see the Pods and Services Reference, on page 65 chapter in this guide.

High Availability of UDP Proxy

The cnSGW-C supports High Availability (HA) of UDP proxy. The HA model of UDP proxy is based on the keepalived virtual IP concepts.

For more information on UDP proxy redundancy, see the High Availability for the UDP Proxy, on page 70 section in the Pods and Services Reference, on page 65 chapter.

Architecture

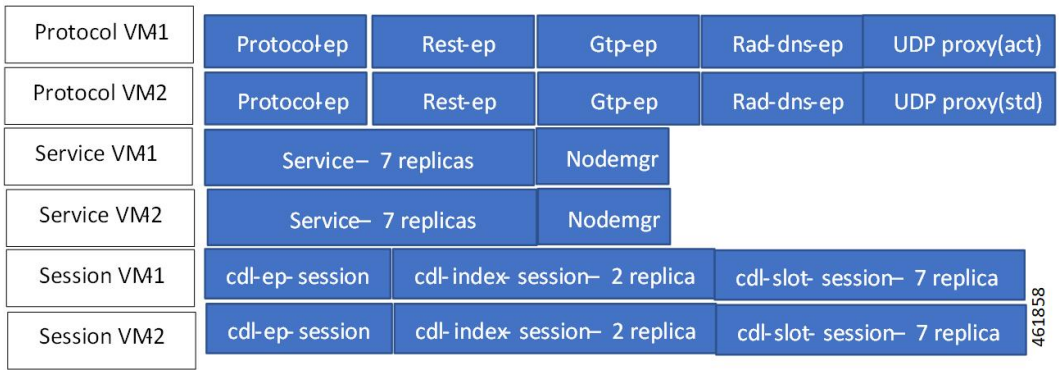
This section describes the recommended layout of cnSGW-C pods and VMs.

cnSGW-C Pod and VM Deployment Layout

This section describes the deployment of cnSGW-C pods and its microservices.

The following figure shows the deployment model of six VMs in cnSGW-C.

Figure 109: VM Deployment Model



In this model, the pods are deployed on VM pairs. Two replicas are available for each protocol pod (for example, rest-ep, protocol-ep, and gtp-ep). One instance is deployed on each protocol VM.

Similarly, service pods and session pods are distributed equally on both the service and session VMs. Such a distribution is controlled by labelling the VMs as well as implementing the K8 affinity and anti-affinity rules during pod scheduling.

This model ensures that, during VM reboot scenarios, at least 50% of the replicas of each pod type are available to handle user signaling.

Graceful pod restart allows pod to complete ongoing processing within 30 seconds. Abrupt pod restart will affect ongoing transactions without impact to PDU sessions.

How it Works

This section provides information on how the resiliency and HA can be achieved.

The cnSGW-C enables inter-pod communication during the pod failure or restart.

During graceful pod restart:

- Ongoing processing is not impacted.
- New messages are not sent to the pod through Kubernetes service.
- Messages with session affinity continue to be received by the pod.
- Existing call flow expected to complete within 30 seconds.

After pod restart:

- All Prometheus metrics of the pod are reset.
- When internal diagnostics is green, the pod status changes to Ready.
- Pod is ready to process the new messages.

When the cnSGW-C VM reboots or the VM is unavailable:

- All pods on the VM are lost.
- Pods on the other available VM continue processing, thus providing high availability.
- VIP, if present, is switched to the other available node.
- It takes about 5 minutes of the node unreachability for Kubernetes to detect the node as down.
- Pods on the node are thereafter not discoverable through Kubernetes service.

After the pod restarts, pods on the VM are scheduled one after another. This operation is similar to the pod restart.

During the VIP and VM reboot, virtual IP is associated with a single VM. UDP proxy binds to N4 VIP address for communication with UPF. UDP proxy binds to S5 VIP address for communication with cnSGW-C.

Reboot of VM with active VIP causes VIP to switch to the other protocol VM. The active UDP proxy failure causes VIP to switch to other protocol VM.

Before the Subscriber Microservices Infrastructure (SMI) handles the VIP monitoring and switchover, make sure that appropriate VIP configuration is available in the SMI deployer. Also, check if the port is set to 28000 and the host priority is equal.

Configuring Pod-level Labelling and Replicas

The node label is configured on the SMI cluster deployer. For information on the configuration commands, see the Mapping Pods with Node Labels, on page 23 section in the Deploying and Configuring cnSGW-C through Operations Center, on page 21 chapter.

Configuration Example

The following is an example of VM labelling and replica configuration.

```
k8 label protocol-layer key smi.cisco.com/node-type value smf-proto
exit
k8 label service-layer key vm-type value smf-svc
exit
k8 label cdl-layer key smi.cisco.com/node-type value smf-cdl
exit
k8 label oam-layer key smi.cisco.com/node-type value oam
exit

endpoint pfcp
 replicas 1
 nodes    2
exit
endpoint service
 replicas 1
 nodes    2
exit
endpoint protocol
 replicas 1
 nodes    2
 vip-ip 209.165.201.28
exit
endpoint sbi
 replicas 1
 nodes    2
```

Configuration Verification

To verify the configuration, use the following show command:

show running-config instance instance-id *instance_id* **endpoint**

The following is an example output of this show command.

```
show running-config instance instance-id 1 endpoint
instance instance-id 1
 endpoint nodemgr
  replicas 1
  nodes    2
 exit
 endpoint gtp
  replicas 1
  vip-ip 209.165.201.29
 exit
 endpoint pfcp
  replicas                2
  enable-cpu-optimization true
  interface n4
   heartbeat
    interval               0
    retransmission-timeout 3
    max-retransmissions    5
```

```
  exit
 exit
exit
endpoint service
 replicas 2
exit
endpoint protocol
 replicas 1
 vip-ip 209.165.201.29
 end
```

This command output displays the configurations related to multiple endpoints, such as endpoint names, pod replicas, and nodes.

Inter-Rack Redundancy Support

Inter-Rack redundancy support refers to the ability of a system or service to maintain its functionality and availability in the event of a failure or outage in one rack can be mitigated by moving the operations to another rack in the same geo location.

Feature Description

The cnSGW-C5GSA GW supports Inter-Rack redundancy in the active-active mode. The Inter-Rack redundancy is achieved through replication of sessions, configuration, and any other data required for seamless failover and failback of services to the remote rack.

How It Works

cnSGW-C (CNF) can be deployed in the same data center to provide service for a catastrophic failure localized to a rack hosting an cnSGW-C cluster.

Each CNF instance service registers with NRF and S11/S5 for DNS entry for MME/SGW. Local HA redundancy allows instance to achieve rack level redundancy in addition to K8 cluster level failures within same data center or handle locally within same K8 cluster if failed containers are per Type-2 < *n*.

where, *n* is a value. For less than 50% of container failures, HA should handle the failures. For more than 50% of container failures, Inter-rack switchover is triggered.

Overview

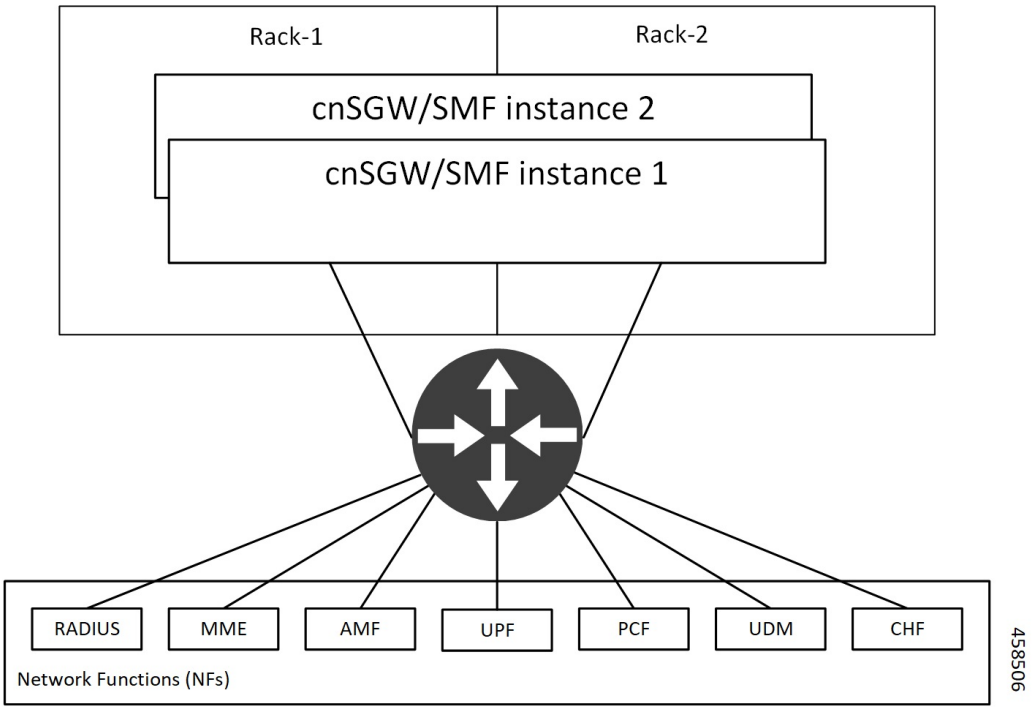
In active-active mode,

- The inter-rack deployment is transparent to the adjacent NFs.
- The inter-rack deployment contains two instances of the CCG function, each instance manifest itself with a set of interface IPs.
- Each instance support sets of sessions and continue to use the same IP for session consistency.
- At a specific time period, one CCG instance can be primary only on one rack and standby on the other rack.
- The set of interface IPs that are associated with the CCG instance, dynamically route to the primary rack of the instance.

cnSGW-C supports primary/standby redundancy in which data is replicated from the primary to standby instance. The primary instance provides services in normal operation. If the primary instance fails, the standby instance becomes the primary and takes over the operation. To achieve inter-rack redundancy, two primary/standby pairs can be set up where each rack is actively processing traffic and standby is acting as backup for the remote rack.

In an Active-Active inter-rack redundancy deployment, consider there are two racks: Rack-1 and Rack-2 located in the same data center. All the NFs are trying to reach instance-1 and instance-2.

Figure 110: Active-Active Inter-Rack Redundancy Deployment



For NFs, both the instances are active. But in real, instance-1 and instance-2 are divided across racks.

Rack-1 has instance-1 and instance-2. In a pre-trigger scenario, instance-1 is local and acts as Primary and instance-2 is in Standby mode.

Rack-2 also has instance-1 and instance-2. In a pre-trigger scenario, instance-2 is local and acts as Primary and instance-1 is in Standby mode.

In case, if Rack-1 goes down, the traffic moves to Rack-2. On Rack-2 both the instances, instance-1 and instance-2 acts as Primary.

Inter-Rack Redundancy Triggers

Inter-rack redundancy supports the following triggers:

- **CLI-based Switchover:** Manual CLI commands are used to switch the roles and trigger inter-rack redundancy failover.
- **BFD Link Failover Detection:** When both the BFD links between the connected rack and the leafs are down, inter-rack redundancy failover is triggered.

- **Local Rack POD Failure Detection:** When threshold percentage of POD replica-sets failing is greater than the configured threshold value, the inter-rack redundancy failover is triggered.
- **Remote Rack POD Failure Detection:** When the remote POD monitoring detects failure breaching threshold percentage, the POD becomes self-primary for that instance.
- **Remote Rack Role Monitoring:** When the remote role monitoring detects that the rack is in Standy_error state, it becomes self-primary.
- **Multi-Compute Failure:** When two or more servers are powered down, it triggers inter-rack redundancy failover.

Rack NF Roles

The following is a list of applicable rack NF roles:



Note

- The **Cachepod**/**ETCD** and the **CDL Replication** happen during all the roles mentioned in the following section.
- If the inter-rack links are down or under periodic heartbeat fails, then these inter-rack redundancy triggers get suspended.

- **PRIMARY**: In this role, the rack is in ready state and actively taking traffic for the given instance.
- **STANDBY**: In this role, the rack is in standby mode, ready to take traffic, but not taking traffic for the given instance.
- **STANDBY_ERROR**: In this role, the rack is in problem state, not active, and not ready to take traffic for the given instance.



Note When the instance role is in **STANDBY_ERROR**, data replication gets halted. The command **show georeplication-status** consistently fails under this condition. However, once the instance role gets transitioned to **STANDBY**, data replication resumes automatically, and the command displays the result as **pass**.

- **FAILOVER_INIT**: In this role, the rack has started to fail over and not in condition to take traffic. The buffer time is two seconds for the application to complete their activity.
- **FAILOVER_COMPLETE**: In this role, the rack has completed the failover and attempted to inform the peer rack about the failover for the given instance. The buffer time is two seconds.
- **FAILBACK_STARTED**: In this role, the manual failover gets triggered with delay from a remote rack for the given instance.

For fresh installation, the rack boots-up with the following roles:

- **PRIMARY**: In this role, the rack is in for the local instance (each rack has local **instance-id** configured to identify the local instance). It is recommended not to configure the pods for monitoring during fresh installation. Once the setup is ready, you can configure the pods for monitoring.

- **STANDBY**: In this role, the rack is in for other instances.

For upgrades, the rack boots-up with the following roles:

- **STANDBY_ERROR**: In this role, the rack is for all the instances as moving the traffic post upgrade needs manual intervention.
- **ETCD**: In this role, the rack stores instance roles.

Note The rolling upgrade or the in-service upgrade is not supported.

General Guidelines

Before configuring the inter-rack redundancy deployment, here are some general guidelines:

- Both racks should be on the same software version.
- Both racks should be configured with same configuration.
- Loopback port of Instance 1 and Instance 2 should be different. Else, REST-EP POD would not come up due to K8 IP/Port conflict.
- Respective interface on both the racks should be on the same VLAN. For example, N4 VLAN of Instance1 and Instance2 should be on the same VLAN. Else, there is a route conflict on Kernel while enforcing BGP policies.
- Consult your Cisco Technical Representative to perform the following procedures to make sure proper roles are assigned.

  For more information, see Software Upgrade on GR Pairs, on page 563.
- Post failover, perform the failback manually after ensuring the rack is healthy. Autonomous failback is not supported.

  For more information, see Recovery Procedure, on page 585.
- Use non-bonded interface in BGP speaker PODs for BGP peering.
- BGP peering per Proto node is supported with only two BGP routers/leafs. Considering two Proto nodes, there can be maximum of four BGP neighborships.
- Use bonded interfaces for Service traffic.
- Geo pod uses two VIPs:
  - Internal-VIP for Inter-POD communication (within the rack)
  - External-VIP for Inter-rack Geo pod communication. Configure only on Proto Nodes on L2 Subnet. This is used to communicate across the racks. This node has external connectivity to other rack.
- Geo Internal IP to be reachable to all nodes within the rack.
- Geo External IP:
- CDL/Kafka VIPs: Configure on CDL Labeled Nodes on L2 Subnet.

- Enable LI tapping on both the racks.
- MDF server should be reachable from both the racks.

Instance Awareness

Instance awareness configuration in cnSGW-C helps to distinguish local rack instance and remote rack instance.

Configuring Inter-Rack Redundancy Instance

This configuration is needed to provide a inter-rack redundancy configuration for multiple rack. With instance ID, endpoint configurations should be configured for each rack.

Sample Configuration 1

The following is a sample configuration for endpoint VIP configuration under one instance:

```
config
  instance instance-id gr_instanceId
    endpoint endpoint_name
        vip-ip vip_ip_address
  exit
exit
```

Example:

```
config
instance instance-id 1
 endpoint sbi
  vip-ip 209.165.201.21
 exit
exit
```

Sample Configuration 2

The following is a sample configuration to provide information on system-id, cluster-id and slice-name under an instance:

```
config
  instances instance instance_id
   system-id system_id
   cluster-id cluster_id
   slice-name cdl_slice_name
  exit
exit
```

Example:

```
config
 instances instance 1
  system-id sgw
  cluster-id sgw
  slice-name 1
 exit
exit
```

Note It is recommended to have the same values for *system-id*, *cluster-id* in the instance, and *app-name*, *cluster-name* in deployment.

Configuring Endpoint Instance Awareness

Only two instances can be configured on each local and remote rack, and corresponding endpoints can be instantiated.

A local instance-id is the identity of the local rack irrespective of if the rack is redundant or not.

Local Instance ID Configuration

The local instance is configured using the local-instance command.

```
local-instance instance 1
```

Endpoint configuration must be under instance specified by each unique instance ID.

Endpoint Configuration Example

Following are a few configuration examples.

Note In the following example, *instance-id* "**1**" is a local instance-id, and endpoints configured under it belong to the local rack.

Optionally, remote rack *instance-id* "**2**" can be configured for endpoints belonging to the inter-rack.

```
instance instance-id 1
 endpoint li
  replicas 1
  nodes    2
  vip-ip 209.165.201.6
  vip-ip 209.165.201.13
 exit
 endpoint gtp
  replicas 1
  nodes    2
  retransmission timeout 5 max-retry 4
  vip-ip 209.165.201.6
  vip-ip 209.165.201.4
  interface s5
   echo interval          60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
  interface s2b
   echo interval 60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
 exit
exit
instance instance-id 2
 endpoint li
  replicas 1
```

```
  nodes    2
  vip-ip 209.165.201.6
  vip-ip 209.165.201.13
 exit
exit
endpoint gtp
  replicas 1
  nodes    2
  retransmission timeout 5 max-retry 4
  vip-ip 209.165.201.6
  vip-ip 209.165.201.5
  interface s5
   echo interval 60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
  interface s2b
   echo interval 60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
 exit
exit
```

Configuring Profile cnSGW-C5GSA GW Instance Awareness

Add instance for PGW FQDN corresponding to local and remote instances.

Example

Following is a configuration example.

Note In the following example, *instance-id* "**1**" is a local instance-id, and the cnSGW-C profile configured under it belongs to the local rack.

Optionally, remote rack *instance-id* "**2**" can be configured for FQDN belonging to the inter-rack.

```
profile sgw sgw1
locality LOC1
instances 1 fqdn cisco.com.apn.epc.mnc456.mcc123
instances 2 fqdn cisco.com.apn.epc.mnc567.mcc123
```

Configuring cnSGW-C Endpoint

Endpoint configuration is required only for cnSGW-C.

Example

The following is a configuration example.

Note In the following example, *instance-id* "**1**" is a local instance-id, and endpoints configured under it belong to the local site.

Optionally, remote site *instance-id* "**2**" can be configured for endpoints belonging to the geo-site.

```
instance instance-id 1
 endpoint nodemgr
  replicas 1
  nodes    1
 exit
 endpoint gtp
  replicas 1
  vip-ip 209.165.201.10
  interface s5e
   vip-ip 209.165.201.29
  exit
  interface s11
   vip-ip 209.165.201.29
  exit
 exit
 endpoint pfcp
  replicas 1
  interface sxa
   heartbeat
    interval               0
    retransmission-timeout 5
    max-retransmissions    3
   exit
  exit
exit
 endpoint service
  replicas 1
 exit
 endpoint protocol
  replicas 1
  vip-ip 209.165.201.29
  interface sxa
   vip-ip 209.165.201.29
  exit
 exit
 endpoint sgw-service
  replicas 1
 exit
exit
instance instance-id 2
 endpoint nodemgr
  replicas 1
  nodes    1
 exit
 endpoint gtp
  replicas 1
  vip-ip 209.165.202.150
  interface s5e
   vip-ip 209.165.201.27
  exit
  interface s11
   vip-ip 209.165.201.27
  exit
 exit
 endpoint pfcp
  replicas 1
  interface sxa
   heartbeat
    interval               0
    retransmission-timeout 5
    max-retransmissions    3
   exit
  exit
exit
```

```
 endpoint service
  replicas 1
 exit
 endpoint protocol
  replicas 1
  vip-ip 209.165.201.27
  interface sxa
   vip-ip 209.165.201.27
  exit
 exit
 endpoint sgw-service
  replicas 1
 exit
exit
```

Dynamic Routing

Border Gateway Protocol (BGP) allows you to create loop-free inter-domain routing between autonomous systems (AS). An AS is a set of routers under a single technical administration. The routers can use an Exterior Gateway Protocol to route packets outside the AS. The Dynamic Routing by Using BGP feature enables you to configure the next-hop attribute of a BGP router with alternate local addresses to service IP addresses with priority and routes. The App-Infra BGP speaker pods enable dynamic routing of traffic by using BGP to advertise pod routes to the service VIP.

Key functionality of Dynamic Routing using BGP:

- Advertising Specific Service IPs: BGP advertises individual service IP addresses using a */32* netmask. This enables dynamic routing for each service IP.
- Next-Hop for Ingress Traffic: When BGP advertises a */32* service IP address, it includes a next-hop address. This next-hop is the service interface's IP address. External networks use this address to forward incoming traffic to the specific service IP. This process ensures efficient and dynamic traffic steering.
- Prioritized Routing via MED: Initially, Master1 advertises the service VIP route with a lower MED value (1210) because the VIP is active on it. Master2 also advertises the route but uses a higher MED value (1220) since the VIP is not active there. External BGP peers select the route from Master1 because it has the lower MED, so incoming traffic goes to Master1. For iBGP, BGP speakers use local preference to prioritize routes.
- Dynamic Failover and Traffic Redirection using VIP Monitoring: If the GTP/Protocol pod on Master1 goes down, the VIP becomes unavailable on Master1 and moves to Master2. Both BGP monitors detect this change. Master1's BGP speaker advertises the VIP route with the higher MED (1220), and Master2's BGP speaker advertises it with the lower MED (1210). External nodes receive these updates and re-evaluate their paths. Incoming traffic then redirects seamlessly to Master2, enabling smooth failover.
- Dynamic Route Learning and Kernel Integration: BGP dynamically learns network prefixes and their next-hop IP addresses from other BGP peers. The learned routes are installed into the kernel routing table. This makes them available for forwarding outgoing traffic
- Policy-Driven Route Filtering for Control: BGP peers may advertise many routes, but not all are accepted or installed in the local kernel routing table. BGP implementation uses routing policies to filter incoming BGP advertisements. This selective acceptance allows BGP speaker to control which IP prefixes the network learns and uses.
- Automatic Route Re-installation on Interface Recovery: If a network interface goes down, the system removes all dependent routes from the kernel routing table. BGP speaker continuously monitors network

interfaces. When a previously down interface returns to service, the BGP speaker automatically reinstalls the relevant routes, restoring network connectivity.

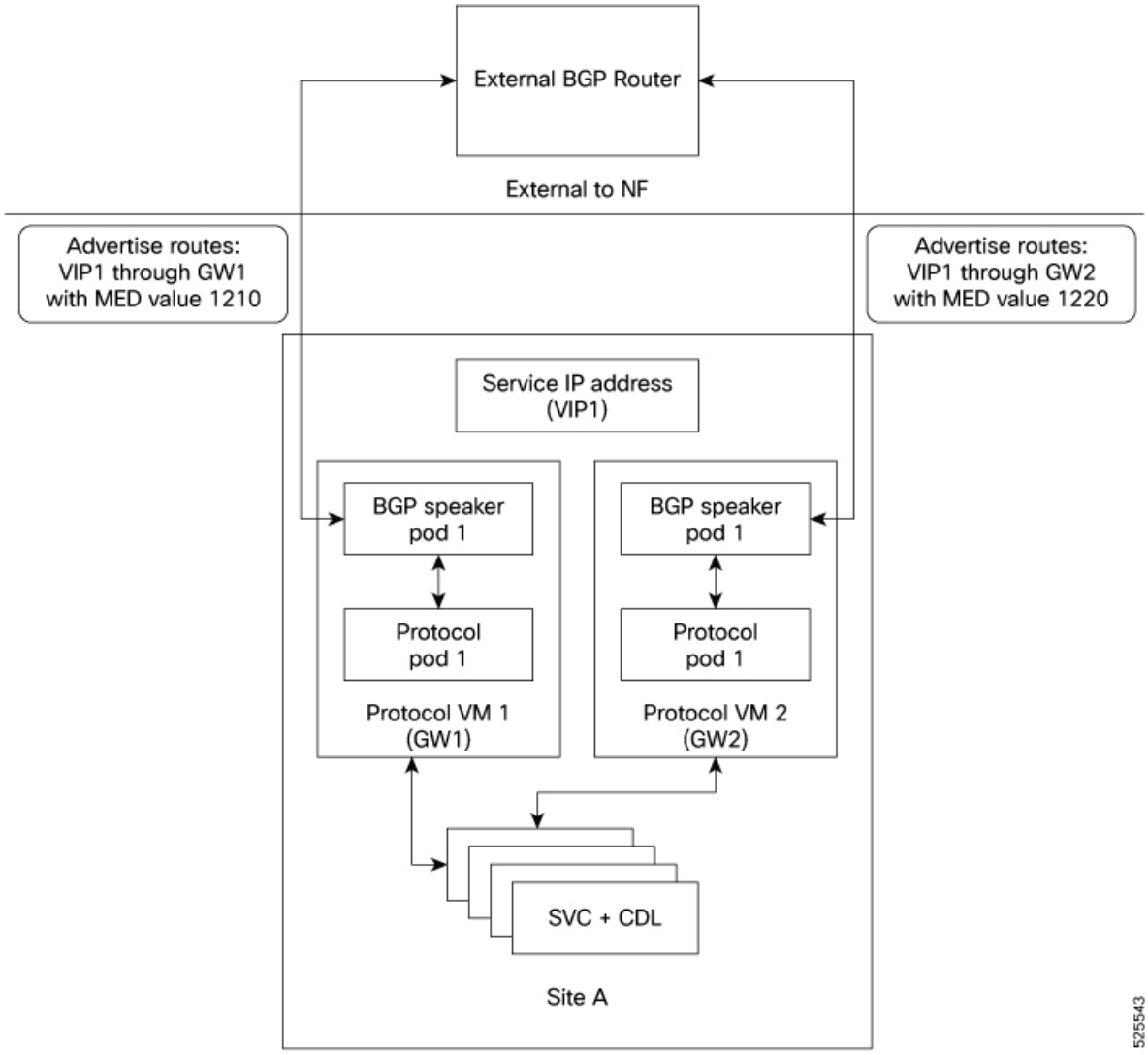
- Policy-Driven Static Route Configuration: BGP speaker uses specific policies to configure static routes manually. These static routes provide fixed, predefined paths to destinations.
- High Availability through Bonded Interfaces and Service IP: The BGP speaker's service IP is assigned to bonded interfaces for redundancy. Each bond includes multiple physical interfaces in an active/standby setup. Each physical interface connects to a BGP router, ensuring highly available network paths for the BGP speaker.
- BGP Peering-Triggered Bond Failover: If BGP peering sessions on the active interface go down, the BGP speaker initiates a bond switchover. This promotes a standby interface to active status. Traffic then redirects seamlessly, maintaining continuous BGP communication.
- Debugging and Troubleshooting through KPI & Log Messages: BGP speakers provide comprehensive statistics and Key Performance Indicators (KPIs) for monitoring. They also generate detailed log messages, which are crucial for debugging and troubleshooting BGP issues.
- BGP MED-Driven Geo-Active/Standby Deployments: In a multi-HA setup (for example, GEO Rack 1 is active and GEO Rack 2 is standby), BGP speakers on active Rack 1 advertise service IPs with lower MED values (1210 and 1220). Speakers on standby Rack 2 use higher MED values (2210 and 2220). BGP's preference for lower MED values directs all traffic to the active rack.
- Monitoring GEO Roles: BGP continuously monitors the active and standby roles of GEO racks. When a role changes, BGP dynamically re-advertises service IP routes with updated MED values. This ensures traffic always goes to the currently active BGP speaker.
- BGP Speaker Status Management: BGP speaker pods report their status to a central Geo pod via ETCD. The status depends on the health of their BGP peerings. If a BGP speaker pod loses all peerings, it is marked as "down" and isolated. Traffic then shifts automatically to the other High Availability (HA) BGP speaker pod. If both HA BGP speaker pods in a GEO rack lose all peerings, the entire rack is marked as "down" in ETCD and set to "standby" by the Geo pod.
- BGP Speaker Pod Architecture with BFD for Accelerated Failover: Each BGP speaker pod contains two containers: a BGP speaker container for routing protocol operations, and a Bidirectional Forwarding Detection (BFD) container. The BFD container quickly detects link or peer failures. This rapid detection allows BGP to respond almost instantly to connectivity issues, reducing failover times and improving network resilience.

Incoming Traffic

BGP uses TCP as the transport protocol, on port 179. Two BGP routers form a TCP connection between one another. These routers are peer routers. The peer routers exchange messages to open and confirm the connection parameters.

The BGP speaker publishes routing information of the protocol pod for incoming traffic in the active standby mode. Use the following image as an example to understand the dynamic routing functionality. There are two protocol pods, pod1 and pod2. Pod1 is active and pod2 is in the standby mode. The service IP address vip1 is configured on both the nodes, on host IP1 and host IP2. BGP pod1 is running on host IP1 and BGP pod2 on host IP2. The host IP address exposes the pod services. BGP speaker publishes the route vip1 through host IP1 and host IP2. It also publishes the preference values, 110 and 100 to determine the priority of pods.

Figure 111: Dynamic Routing for Incoming Traffic in the Active-standby Topology



For high availability, each cluster has two BGP speaker pods with Active-standby topology. Kernel route modification is done at host network level where the protocol pod runs.

MED Value

The Local Preference is used only for IGP neighbours, whereas the MED Attribute is used only for EGP neighbours. A lower MED value is the preferred choice for BGP.

Table 199: MED Value

Bonding Interface Active	VIP Present	MED Value	Local Preference
Yes	Yes	1210	2220
Yes	No	1220	2210

Bonding Interface Active	VIP Present	MED Value	Local Preference
No	Yes	1215	2215
No	No	1225	2205

Bootstrap of BGP Speaker Pods

The following sequence of steps set up the BGP speaker pods:

1. The BGP speaker pods use TCP as the transport protocol, on port 179. These pods use the AS number configured in the Ops Center CLI.
2. Register the Topology manager.
3. Select the Leader pod. The Active speaker pod is the default choice.
4. Establish connection to all the BGP peers provided by the Ops Center CLI.
5. Publish all existing routes from ETCD.
6. Configure import policies for routing by using CLI configuration.
7. Start gRPC stream server on both the speaker pods.
8. Similar to the cache pod, two BGP speaker pods must run on each Namespace.

For more information on Dynamic Routing, see the *Dynamic Routing by Using BGP* chapter in the *UCC Serving Gateway Control Plane Function - Configuration and Administration Guide*.

Configuring Dynamic Routing Using BGP

This section describes how to configure the dynamic routing using BGP.

Configuring AS and BGP Router IP Address

To configure the AS and IP address for the BGP router, use the following commands:

```
config
   router bgp local_as_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.

  In a inter-rack redundancy deployment, you need to configure two Autonomous Systems (AS).

  - One AS for leaf and spine.
  - Second AS for both racks: Rack-1 and Rack-2.

Configure Router ID (Optional)

By default, the BGP speaker uses the IPv4 address of the BGP server running on an interface as the Router ID. BGP speaker uses this IPv4 address as the Router ID for BGP peering for both IPv4 BGP servers and Dual-mode BGP servers.

To configure a custom Router ID, assign an IPv4 address to the loopback (lo) interface on the node where the BGP Speaker pod is running. This assigned IPv4 address is then used as the Router ID.



Note Configuring a Router ID is mandatory if you plan to run the BGP server exclusively with IPv6 addresses (i.e., without any IPv4 address on the interface where the BGP server is running).

Configuring BGP Service Listening IP Address

To configure the BGP service listening IP address, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.

Configuring BGP Neighbors

To configure the BGP neighbors, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
      neighbor neighbor_ip_address remote-as as_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.
- **neighbor** *neighbor_ip_address*—Specify the IP address of the neighbor BGP router.
- **remote-as** *as_number*—Specify the identification number for the AS.

Configuring Bonding Interface

To configure the bonding interface related to the interfaces, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
      bondingInterface interface_name
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.
- **bondingInterface** *interface_name*—Specify the related bonding interface for an interface. If the bonding interface is active, then the BGP gives a higher preference to the interface-service by providing a lower MED value.

Configuring Learn Default Route

If the user configures specific routes on their system and they need to support all routes, then they must set the **learnDefaultRoute** as **true**.

Note This configuration is optional.

To configure the Learn Default Route, use the following commands:

```
config
   router bgp local_as_number
      learnDefaultRoute true/false
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **learnDefaultRoute** *true/false*—Specify the option to enable or disable the **learnDefaultRoute** parameter. When set to true, BGP learns default route and adds it in the kernel space. By default, it is false.

Configuring BGP Port

To configure the Port number for a BGP service, use the following commands:

```
config
   router bgp local_as_number
      loopbackPort port_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **loopbackPort** *port_number*—Specify the port number for the BGP service. The default value is 179.

Policy Addition

The BGP speaker pods learns many route information from its neighbors. However, only a few of them are used for supporting the outgoing traffic. This is required for egress traffic handling only, when cnSGW-C is sending information outside to AMF/PCF. Routes are filtered by configuring import policies on the BGP speakers and is used to send learned routes to the protocol pods.

A sample CLI code for policy addition and the corresponding descriptions for the parameters are shown below.

```
$bgp policy <policy_Name> ip-prefix 209.165.200.225 subnet 16 masklength-range 21..24
as-path-set "^65100"
```

Table 200: Import Policies Parameters

Element	Description	Example	Optional
as-path-set	AS path value	"^65100"	Yes
ip-prefix	Prefix value	"209.165.200.225/16"	Yes
masklength-range	Range of length	"21..24"	Yes
interface	Interface to set as source IP (default is VM IP)	eth0	Yes
gateWay	Change gateway of incoming route	209.165.201.30	Yes
modifySourceIp	Modify source ip of incoming route Default value is False.	true	Yes
isStaticRoute	Flag to add static IP address into kernel route Default value is False.	true	Yes

Configuring BGP Speaker

This configuration controls the number of BGP speaker pods in deployment. BGP speaker advertises service IP information for incoming traffic from both the racks.

Note

- Use non-bonded interface in BGP speaker pods for BGP peering.
- BGP peering per Proto node is supported with only two BGP routers/leafs. Considering two Proto nodes, there can be maximum of four BGP neighborships.

```
instance instance-id  instance_id endpoint bgpspeaker interface { bgp | bfd
 } internal base-port start base_port_number
```

```
config
instance instance-id instance_id
 endpoint bgpspeaker
   replicas replica_id
```

```
   nodes node_id
   interface bgp
     internal base-port start base_port_number
   exit
   interface bfd
     internal base-port start base_port_number
  exit
exit
```

NOTES:

- **instance instance-id** *instance_id*—Specify the GR instance ID.
- *base_port_number*—Specify the port range only if logical NF is configured. This range depends on your deployment.

Example

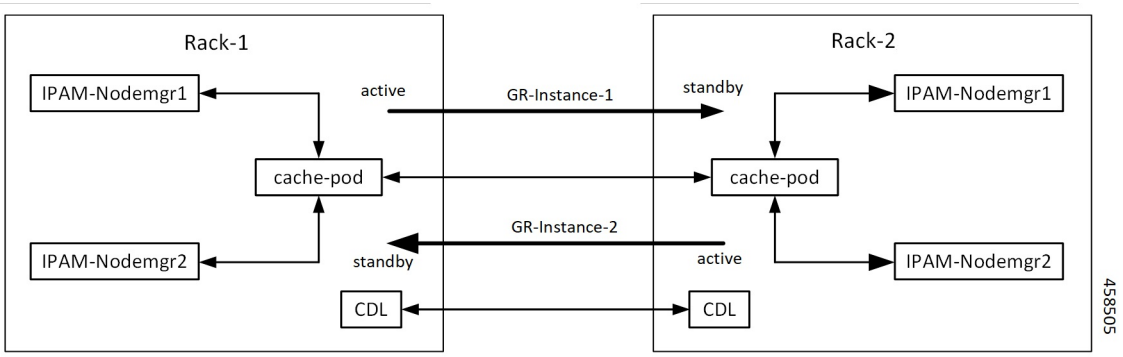
The following is a configuration example:

```
instance instance-id 1
endpoint bgpspeaker
   replicas 1
   nodes    2
   interface bgp
     internal base-port start {24000}
   exit
   interface bfd
     internal base-port start {25000}
 exit
```

IPAM

This section describes IP Address Management (IPAM) at the rack level.

Figure 112: IPAM



During UPF registration, active IPAM instance reserves four address-ranges per UPF per DNN.

- Range-1: Active cluster, nodemgr-1
- Range-2: Active cluster, nodemgr-2
- Range-3: Standby cluster, nodemgr-1

- Range-4: Standby cluster, nodemgr-2

During normal operation, Rack-1 handles UPF-register/release, address-allocate/release for subscribers coming up in GR-instance-1.

If Rack-2 goes down, Rack-1 gets role-change trigger for GR-Instance-2.

- IPAM in Rack-1, restores the content of GR-Instance-2 from local-cache-pod (which was already synced)
- IPAM in Rack-1 handles UPF-Register/Release and address-allocate/release for subscribers coming up with GR-Instance-2 using the restored content in addition to handling GR-Instance-1.

Each IPAM pool is associated to a GR-Instance, with the following:

- Pool name is unique across all the instances.
- Address-ranges are unique within VRF and across all the instances.

The same pool configuration must be configured in both the active and standby cnSGW-C clusters of a particular instance.

During address-allocation, active instance assign free-IP from reserved address-range for the UPF.

Incase new address-ranges is not available, change ownership of standby's address-range to current active instance and continue assigning address-ranges from it.

Configuring IPAM

The following section provides IPAM configuraton examples.

cnSGW-C-1 Example

The following is a configuration example for cnSGW-C-1:

```
ipam
 instance 1
  address-pool pool-1
  vrf-name ISP
   tags
   dnn dnn-1
  exit
  ipv4
    address-range 209.165.201.1 209.165.201.31
 exit
instance 2
 address-pool pool-2
  vrf-name ISP
  tags
   dnn dnn-2
  exit
  ipv4
    address-range 209.165.202.129 209.165.202.159
 exit
exit
```

cnSGW-C-2 Example

The following is a configuration example for cnSGW-C-2:

```
ipam
 instance 1
```

```
 address-pool pool-1
  vrf-name ISP
  tags
   dnn dnn-1
  exit
  ipv4
    address-range 209.165.201.1 209.165.201.31
 exit
 instance 2
 address-pool pool-2
  vrf-name ISP
   tags
   dnn dnn-2
  exit
  ipv4
    address-range 209.165.202.129 209.165.202.159
 exit
exit
```

Geo Replication

The Geo-replication is used in inter-rack communication and for POD or VIP or BFD monitoring within the rack. The Geographic Redundancy comprises with the following:

- Two instances of Geo pods are running for each rack.
- Two Geo pods functions in Active-Standby mode.
- Each Geo pod instance is spawned on a different Proto node or VM.
- Geo pod running on the Proto node or VM having VIP is Active Geo pod.
- In the event of Active Geo pod restart, VIPs get switched to other Proto node or VM and Standby Geo pod running on the other Proto node/VM becomes active.
- Geo pod uses host networking mode (similar to UDP-Proxy).
- Geo pod uses two VIPs:
  - **Internal**: VIP for Inter-POD communication (within the rack)
  - **External**: VIP for Inter-rack Geo pod communication

    It configures only on Proto Nodes on the L2 Subnet. It's used to communicate across the racks. This node has external connectivity to other Rack.
- Logical-NF-InstanceID must be configured same for both cnSGW-Cs in GR-Pair.
- For KeepAliveD monitoring:
  - Geo pod uses base port as: 15000+(Logical-NF-InstanceID * 32) + 4

    Geo pod base port must be different than BGP speaker pod port.
    - The default port (without logical cnSGW-C) as: 15004
    - For Logical cnSGW-C configured with logical-nf-instance-id as 1, and then the port as: 15036
  - UDP-Proxy pod uses base port as: 28000+Logical-NF-InstanceID.
    - The default port (without logical cnSGW-C) as: 28000

- For Logical cnSGW-C configured with logical-nf-instance-id as 1, and then the port as: 28001

- BGPSpeaker-pod uses default base port as: 20000+(Logical-NF-InstanceID * 32)+4.
  - The default port (without logical cnSGW-C) as: 20004
  - For logically cnSGW-C configured with logical-nf-instance-id as 1, and then the port as: 20036



Note Only ETCD and cache pod data gets replicated to the standby rack.

Configuring ETCD/Cachepod Replication

Endpoints must be configured under an instance. Two Geo-Redundancy pods are needed on each rack. You should also configure VIP for internal and external Geo interface for ETCD/CachePod replication.

instance instance-id *instance_id* **endpoint geo interface { geo-internal | geo-external } vip-ip {** *vip_ip_address* **} vip-port {** *vip_port_number* **}**

```
config
instance instance-id instance_id
 endpoint geo
  replicas replica_id
  nodes node_id
  internal base-port start base_port_number
  interface geo-internal
   vip-ip vip_ip_address vip-port vip_port_number
  exit
  interface geo-external
   vip-ip vip_ip_address vip-port vip_port_number
  exit
exit
exit
```

NOTES:

- **instance instance-id** *instance_id*—Specify GR instance ID. One instance ID for local rack and other for another rack.
- **vip-ip** *vip_ip_address*—Specify VIP IP address for Internal/External Geo interface.
- **vip-port** *vip_port_number*—Specify VIP port number.
- **internal base-port start** *base_port_number*—Specify port range only if logical NF is configured.

Example

The following is a configuration example:

```
instance instance-id 1
endpoint geo
  replicas 1
  nodes 2
```

```
  internal base-port start 25000
  interface geo-internal
   vip-ip 209.165.201.8 vip-port 7001
  exit
  interface geo-external
   vip-ip 209.165.201.8 vip-port 7002
  exit
exit
```

Geo Monitoring

This section describes Geo monitoring.

Pod monitoring

Table 201: Feature history

Feature name	Release information	Feature description
Pod monitoring in intra-rack geo redundancy	2026.02.0	This feature enables pod monitoring of all the app-infra pods, enabling enhanced failure detection and automatic failover. When the configured threshold percentage of pod replicas fail, the Geo pod initiates a switchover to the peer rack to maintain service continuity.

The geo pod monitors all the pods that are based on app-infra library. Upon detecting a pod failure for a certain configured percentage, the geo pod switches over to peer rack.

Remote Cluster Monitoring

Remote cluster monitoring auto corrects roles (it becomes self-primary, when the remote rack is in **STANDBY_ERROR** state) for uninterrupted traffic flow of traffic. However, this auto role correction gets done only for specific roles.

To configure this feature, use the following sample configuration:

```
config
   geomonitor
      remoteclustermonitor
         retryCount value
         retryInterval interval_value
         end
```

NOTES:

- **retryCount** *value*—Specify the retry count before making the current rack **PRIMARY**. It should be an integer in the range of 1-10. The default value is 3.
- **retryInterval** *interval_value*—Specify the retry interval in the count of milliseconds, after which the remote rack status gets fetched. It should be an integer in the range of 200-50000. The default value is 3000.

Configuration Example

The following is an example configuration

```
geomonitor remoteclustermonitor
retryCount 3
retryInterval 3000
```

Traffic monitoring

Table 202: Feature history

Feature name	Release information	Feature description
Traffic monitoring in intra-rack geo redundancy	2026.02.0	The traffic monitoring capability prevents the service outages during the failure scenarios. This feature allows the geo-replication pod to monitor the traffic on the STANDBY instance.

In a Geo-redundancy deployment for the 5GSA GW, maintaining high availability is critical. The traffic monitoring feature prevents service outages and ensures smooth flow of traffic during failure scenarios.

BFD Monitoring

Bidirectional Forwarding Detection (BFD) protocol is used for Faster Network Failure Detection along with BGP. Whenever connectivity between BGP peering fails with cluster (NF), failover is triggered to minimize traffic failure impact.

```
config
  router bgp as
    bfd interval interval min_rx min_rx multiplier multiplier
    loopbackPort loopbackPort loopbackBFDPort loopbackBFDPort
  interface interface_id (BGP on non-bonded interface <-- loopbackEth)
    bondingInterface bondingInterface (leaf6-nic)
    bondingInterface bondingInterface (leaf6-nic)
    neighbor neighbor_ip_address remote-as remote_as fail-over fail_over_type
  exit
  interface interface_id (BGP on non-bonded interface <-- loopbackEth)
    bondingInterface bondingInterface (leaf7-nic)
    bondingInterface bondingInterface (leaf7-nic)
    neighbor bondingInterface remote-as remote_as fail-over fail_over_type
  exit
  policy-name policy_name
   as-path-set as_path_set
   gateWay gateWay_address
   interface interface_id_source
   ip-prefix ip_prefix_value
   isStaticRoute false | true
   mask-range mask_range
   modifySourceIp false | true
```

```
  exit
exit
```

NOTES:

- **bgp** *as*—Specify the Autonomous System (AS) path set.
- **bfd**—Specify BFD configuration.
  - **interval** *interval* —Specify BFD interval in milliseconds.
  - **min_rx** *min_rx*—Specify BFD minimum RX in milliseconds.
  - **multiplier** *multiplier*—Specify BFD interval multiplier.
- **interface** *interface_id*—Specify BGP local interface.
  - **bondingInterface** *bondingInterface*—Specify linked bonding interface.
  - **neighbor** *neighbor_ip_address*—Specify IP address of neighbor.
    - **fail-over** *fail_over_type*—Specify failover type.
    - **remote-as** *remote_as*—Specify Autonomous System (AS) number of BGP neighbor.
- **learnDefaultRoute**—Learn default route and add it in kernel space
- **loopbackBFDPort** *loopbackBFDPort*—Specify BFD local port.
- **loopbackPort** *loopbackPort*—Specify BGP local port.
- **policy-name** *policy_name*—Specify policy name.
  - **as-path-set** *as_path_set*—Specify Autonomous System (AS) path set.
  - **gateWay** *gateWay_address*—Specify gateway address.
  - **interface** *interface_id_source*—Specify interface to set as source IP.
  - **ip-prefix** *ip_prefix_value*—Specify IP prefix value.
  - **isStaticRoute** *false | true*—Specify whether to add static route in kernel space. Default value is false.
  - **mask-range** *mask_range*—Specify mask range.
  - **modifySourceIp** *false | true*—Modify source IP of the incoming route. Default value is false.

    true: This option is used for non-UDP related VIPs. Source IP of the given interface is used as Source IP while sending out packets from cnSGW-C.

    false: This option is used for all UDP related VIPs. VIP is used as Source IP while sending out packets from cnSGW-C.

Example

Following are configuration examples:

```
router bgp 65000
  bfd interval 250000 min_rx 250000 multiplier 3
```

```
  loopbackPort 179 loopbackBFDPort 3784
interface ens160 (BGP on non-bonded interface <-- loopbackEth)
  bondingInterface enp216s0f0 (leaf6-nic)
  bondingInterface enp216s0f1 (leaf6-nic)
  neighbor leaf6-ip remote-as 60000 fail-over bfd
exit
interface ens192 (BGP on non-bonded interface <-- loopbackEth)
  bondingInterface enp94s0f1 (leaf7-nic)
  bondingInterface enp94s0f0 (leaf7-nic)
  neighbor leaf7-ip remote-as 60000 fail-over bfd
exit
policy-name allow-all ip-prefix 209.165.201.30/0 mask-range 0...32
exit
```

BGP router configuration with BFD

```
show running-config router
router bgp 65142
 learnDefaultRoute false
 bfd interval 250000 min_rx 250000 multiplier 3
 interface enp94s0f0.3921
  bondingInterface enp216s0f0
  bondingInterface enp94s0f0
  neighbor 209.165.201.24 remote-as 65141 fail-over bfd
 exit
 interface enp94s0f1.3922
  bondingInterface enp216s0f1
  bondingInterface enp94s0f1
  neighbor 209.165.202.24 remote-as 65141 fail-over bfd
```

Show BFD status of neighbor

```
show bfd-neigbor
status-details

----- bgpspeaker-pod-1----

Peer             Status

209.165.202.142  STATE_DOWN
----- bgpspeaker-pod-2----

Peer             Status

209.165.202.142  STATE_UP
```

```
policy-name allow-n11 ip-prefix 209.165.200.225/54 mask-range 25..32 interface bd1.n11.2271
 modifySourceIp true isStaticRoute true gateWay 209.165.201.14
```

In the above example, *modifySourceIp* is set to true.

- AMF subnet: 209.165.200.225/54

  N11 Svc Bonded Physical Interface: bd1.n11.2271 (IP address - 209.165.201.23)

  N11 Svc Bonded VxLAN Anycast GW: 209.165.201.14

  N11 VIP Address: 209.165.201.7

- cnSGW-C Outbound Packet (will have source IP as 209.165.201.23)

  Inbound Packet to cnSGW-C (will have destination IP as 209.165.201.7)

```
policy-name allow-n4-1 ip-prefix 209.165.201.17/41 mask-range 24..32 interface bd2.n4.2274
 gateWay 209.165.201.17
```

In the above example, *modifySourceIp* is set to false (default).

- UPF N4 Interface IP: 209.165.201.17/41

  N4 Svc Bonded Physical Interface: bd2.n4.2274 (IP address - 209.165.201.23)

  N4 Svc Bonded VxLAN Anycast GW: 209.165.201.17

  N4 VIP Address: 209.165.201.14

- cnSGW-C Outbound Packet (will have source IP as 209.165.201.14)

  Inbound Packet to cnSGW-C (will have destination IP as 209.165.201.14)

CDL GR Deployment

By default, CDL is deployed with two replicas for db-ep, 1 slot map (2 replicas per map), and 1 index map (2 replicas per map).

Note It is recommended to configure the CDL container in YANG.

Prerequisites for CDL GR

Before deploying the CDL GR, user must configure the following:

- CDL Session Database and define the base configuration.
- Kafka for CDL.
- Zookeeper for CDL.

CDL Instance Awareness and Replication

In CDL, along with existing GR related parameters, GR instance awareness must be enabled using a feature flag on all the racks. Also, the mapping of system-id to slice names should also be provided for this feature to work on all the racks.

The CDL is also equipped with Geo Replication (GR) failover notifications, which can notify the timer expiry of session data and bulk notifications to the currently active rack. The CDL uses Border Gateway Protocol (BGP) through App-Infra for the GR failover notifications.

The CDL subscribes to the key value on both the GR racks. The App-Infra sends notifications to the CDL when there is any change in these key values. A key value indicates the state of the CDL System ID or the GR instance. The GR instance is mapped to the CDL slices using the CDL system ID or the GR instance ID in the key.

The system ID is mandatory on both the racks. The GR instance ID in the NF configuration must match the CDL system ID.

CDL has instance-specific data slices. It also allows users to configure instance-specific slice information at the time of bringing up.

- CDL notifies the data on expiry or upon bulk notification request from the active slices.

- CDL determines the active instance based on the notification from app-infra memory-cache.
- CDL slice is a partition within a CDL instance to store a different kind of data. In this case, NF stores a different instance of data.

Note CDL slice name should match with the slice-name configured in GR.

Configuring CDL Instance Awareness

The following command is used to configure CDL instance awareness.

```
config
cdl
 datastore datastore_session_name
   features
    instance-aware-notification
     enable [ true | false ]
     system-id system_id
      slice-names slice_names
   end
```

NOTES:

- **datastore** *datastore_session_name*—Specify the datastore name.
- **enable [true | false]**—Enables the GR instance state check for slices.
- **system-id** *system_id*—Mapping of system ID to slice name.
- **slice-names** *slice_names*—Specify the list of slice names associated with the system ID. CDL slice name should match with the slice-name configured in GR.

Example

The following is a configuration example:

```
cdl datastore session
 features instance-aware-notification enable true
 features instance-aware-notification system-id 1
  slice-names [ sgw1 smf1 ]
 exit
 features instance-aware-notification system-id 2
  slice-names [ sgw2 smf2 ]
 end
```

Configuring CDL Replication

This section describes CDL replication configuration.

1. Configure Rack-1 CDL HA system without any Geo-HA-related configuration parameters.
   a. Set the System ID as 1 in the configuration.
   b. Set the slot map/replica and index map/replica and Kafka replica as per requirements.

The following is a sample configuration:

```
cdl system-id 1
cdl node-type session
cdl datastore session
endpoint replica replica_id
  slot map 4
  slot replica 2
  index map 1
  index replica 2
cdl kafka replica 2
```

1. Configure external IPs on Rack-1 for Rack-2 to Rack-1 communication.
   a. Enable geo-replication on Rack-1 and configure the remote Rack as 2 for Rack-1.

      cdl enable-geo-replication *true*
   b. Configure the external IP for CDL endpoint to be accessed by Rack-2.

      cdl datastore session endpoint external-ip *site-1_external_ip*
   c. Configure the external IP and port for all Kafka replicas.

      So, if two replicas (default) are configured for Kafka, user need to provide two different <*ip*>+<*port*> pairs.

      cdl kafka external-ip *site-1_external_ip port1* **cdl kafka external-ip** *site-1_external_ip port2*

2. Add remote rack information on Rack-2.
   - Remote rack cdl-ep configuration on Rack-2:

     cdl remote-site 1 db-endpoint host *site-1_cdl_ep_ip*

     cdl remote-site 1 db-endpoint port *site-1_cdl_ep_port*

     (Port Example: 8882)
   - Remote rack Kafka configuration on Rack-2:

     ```
     cdl remote-site 1 kafka-server site-1_kafka1_ip site-1_kafka1_port
     ```

     ```
     cdl remote-site 1 kafka-server site-1_kafka2_ip site-1_kafka2_port
     ```
   - Direct the session datastore configuration to remote Rack-2 configuration:

     ```
     cdl datastore session geo-remote-site 1
     ```
   - (Optional) Configure the SSL certificates to establish a secure connection with remote rack on Rack-1. All the certificates are in multi-line raw text format. If the certificates are not valid, the server continues with non-secure connection.

     cdl ssl-config certs *site-2_external_ip* **ssl-key** *<ssl_key>*

     cdl ssl-config certs *site-2_external_ip* **ssl-crt** *<ssl_crt>*

3. Commit GR configuration on Rack-2:
   - Commit the configuration and let the pods be deployed on Rack-2.

- Verify all pods are in running state.
- Once both the racks are deployed, verify that the mirror maker pods on both racks are running and in ready state.

Examples

HA:

```
cdl node-type db-ims

cdl datastore session
 endpoint replica 2
 index map    1
 index write-factor 1
 slot replica 2
 slot map     4
 slot write-factor 1
exit

k8 label cdl-layer key smi.cisco.com/node-type value smf-ims-session
```

Rack-1:

```
cdl system-id          1
cdl node-type          session
cdl enable-geo-replication true
cdl zookeeper replica 1

cdl remote-site 2
 db-endpoint host 209.165.201.21 >> Rack-2 external CDL IP
 db-endpoint port 8882
 kafka-server 209.165.201.21 10092 >> Rack-2 external CDL IP
 exit
exit

cdl label-config session
 endpoint key smi.cisco.com/node-type1
 endpoint value smf-cdl
 slot map 1
  key   smi.cisco.com/node-type1
  value smf-cdl
 exit
 index map 1
  key   smi.cisco.com/node-type1
  value smf-cdl
 exit
exit
cdl logging default-log-level debug

cdl datastore session
 label-config    session
 geo-remote-site [ 2 ]
 slice-names     [ 1 2 ]
 endpoint cpu-request 100
 endpoint replica 2
 endpoint external-ip 209.165.201.25 >> Rack-1 external CDL IP
 endpoint external-port 8882
 index cpu-request 100
 index replica 2
 index map   1
 slot cpu-request 100
 slot replica 2
```

```
 slot map    1
exit

cdl kafka replica 1
cdl kafka label-config key smi.cisco.com/node-type1
cdl kafka label-config value smf-cdl
cdl kafka external-ip 209.165.201.25 10092 >> Rack-1 external CDL IP
```

Rack-2:

```
cdl system-id          2
cdl node-type          session
cdl enable-geo-replication true
cdl zookeeper replica 1

cdl remote-site 1
 db-endpoint host 209.165.201.25 >> Rack-1 external CDL IP
 db-endpoint port 8882
 kafka-server 209.165.201.25 10092 >> Rack-1 external CDL IP
 exit
exit

cdl label-config session
 endpoint key smi.cisco.com/node-type12
 endpoint value smf-cdl
 slot map 1
  key   smi.cisco.com/node-type12
  value smf-cdl
 exit
 index map 1
  key   smi.cisco.com/node-type12
  value smf-cdl
 exit
exit

cdl datastore session
 label-config    session
 geo-remote-site [ 1 ]
 slice-names     [ 1 2 ]
 endpoint cpu-request 100
 endpoint replica 2
 endpoint external-ip 209.165.201.21 >> Rack-2 external CDL IP
 endpoint external-port 8882
 index cpu-request 100
 index replica 2
 index map   1
 slot cpu-request 100
 slot replica 2
 slot map    1
exit

cdl kafka replica 1
cdl kafka label-config key smi.cisco.com/node-type12
cdl kafka label-config value smf-cdl
cdl kafka external-ip 209.165.201.21 10092 >> Rack-2 external CDL IP
```

Lawful Intercept

The Lawful Intercept (LI) feature enables law enforcement agencies (LEAs) to intercept subscriber communications. The LI functionality provides the network operator the capability to intercept control and data messages of the targeted mobile users. To invoke this support, the LEA requests the network operator to start the interception of a particular mobile user. Legal approvals support this request.

1. Lawful Intercept (LI) tap should be configured/enabled on all the racks. If LI configuration fails on one rack, LEA should re-configure it so that for a given subscriber tap is enabled on all the racks.



Note LI tap configuration is not synchronized across racks.

Hence, LI tap configuration is mandatory on all the racks.

For more information on LI tap configuration, contact your Cisco Technical Representative.

2. GR instance awareness is applicable for lawful-intercept src-address only.

   Example:

   ```
   lawful-intercept instance 1 src-addr 209.165.200.225
   ```

   OR

   ```
   lawful-intercept
     instance 1
       src-addr 209.165.200.225
   ```

3. `show` commands are not instance-aware. It shows all the taps configured in a given cluster.

   For more information on LI `show` commands, contact your Cisco Technical Representative.

4. In case all GR instances are in Standby state in a cluster and active LI tap fails with CLI message `Rack is in standby mode, Active Tap is not allowed. Try camp on`, configure camp-on tap for the same subscriber.

RADIUS Configuration

NAS-IP and NAS-Identifier is instance-aware. You can configure different NAS-IP and NAS-Identifier per instance-id in profile-radius configuration. Existing non-instance based NAS-IP and NAS-Identifier configuration is used as default nas-ip and default nas-id for local instance of the rack.

Example

Following are a few configuration examples.

```
profile radius
 attribute
  instance 1
   nas-ip 209.165.200.225  --> Instance-1 specific NAS-IP, used for common AUTH & ACCT
   nas-identifier  smf1    --> Instance-1 specific NAS-Identifier, used for common AUTH &
ACCT
  exit
  instance 2
   nas-ip 209.165.200.230  --> Instance-2 specific NAS-IP, used for common AUTH & ACCT
   nas-identifier  smf2    --> Instance-2 specific NAS-Identifier, used for common AUTH &
ACCT
  exit
 exit
 accounting
  attribute
   instance 1
    nas-ip 209.165.200.225  --> Instance-1 specific NAS-IP, used for common ACCT
    nas-identifier  smf1    --> Instance-1 specific NAS-Identifier , used for common ACCT
   exit
```

```
   instance 2
    nas-ip 209.165.200.230  --> Instance-2 specific NAS-IP, used for common ACCT
    nas-identifier  smf2    --> Instance-2 specific NAS-Identifier , used for common ACCT
   exit
  exit
 exit
 server-group g1
  attribute
   instance 1
    nas-ip 209.165.200.225  --> Instance-1 specific NAS-IP, used for server-group <g1> AUTH
 & ACCT
    nas-identifier  smf1    --> Instance-1 specific NAS-ID, used for server-group <g1> Auth
 &Acct
   exit
   instance 2
    nas-ip 209.165.200.230  --> Instance-2 specific NAS-IP, used for server-group <g1> AUTH
 & ACCT
    nas-identifier  smf2    --> Instance-2 specific NAS-ID,used for server-group <g1>AUTH&ACCT

   exit
  exit
  accounting
   attribute
    instance 1
     nas-ip 209.165.200.225 --> Instance-1 specific NAS-IP, used for server-group <g1> ACCT

     nas-identifier  smf1   --> Instance-1 specific NAS-ID, used for server-group <g1> ACCT

    exit
    instance 2
     nas-ip 209.165.200.230 --> Instance-2 specific NAS-IP, used for server-group <g1> ACCT

     nas-identifier  smf2   --> Instance-2 specific NAS-ID, used for server-group <g1> ACCT

    exit
   exit
  exit
 exit
exit
```

Since **endpoint** *pod* configuration is moved under specific instance, Radius Disconnect-Request VIP is also instance-aware.

```
instance instance-id 1
 endpoint radius
  replicas 1
  interface coa-nas
   vip-ip 209.165.202.130 vip-port 3799  --> Instance-1 specific Radius-Disconnect-Msg-VIP
 & PORT
  exit
 exit
exit
instance instance-id 2
 endpoint radius
  replicas 1
  interface coa-nas
   vip-ip 209.165.202.129 vip-port 3799  --> Instance-2 specific Radius-Disconnect-Msg-VIP
 & PORT
  exit
 exit
exit
```

Software Upgrade on GR Pairs

Considering `config commit` as reference. The same checklist is also applicable for other upgrade scenarios.

Checklist

Note Do not perform `cluster sync` on both racks (Rack-1 and Rack-2) at the same time. Trigger manual switchover on Rack-1 before proceeding with Rack-1 upgrade.

- Do not perform `config commits` on both racks at the same time. Perform `config commit` on each rack separately.
- Before to the `config commit` procedure on Rack-1, initiate the CLI-based switchover on Rack-1 and make sure that Rack-2 is having Primary ownership for both the instances (instance-id 1 and instance-id 2).
- Perform `config commit` on Rack-1. Wait for the successful `config commit`, PODs restart, and are back in running state to fetch the latest helm charts (if applicable).
- Revert the role of Rack-1 to be Primary (Switch/Reset roles on both racks).
- Verify that the available roles of Rack-1 (Primary) and Rack-2 (Standby) are on the expected status.
- Repeat the preceding checklist for Rack-2.

Software Upgrade

Upgrading the Rack-1, when the GR is Enabled:

1. Verify that the available roles of both instances on Rack-1 are in PRIMARY/STANDBY.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "STANDBY"
```

2. Initiate switch role for both instances on Rack-1 to STANDBY with failback-interval of 0 seconds. This step transitions the roles from PRIMARY/STANDBY to STANDBY_ERROR/STANDBY_ERROR.

```
geo switch-role instance-id 1 role standby [failback-interval 0]
geo switch-role instance-id 2 role standby [failback-interval 0]
```

Note
- Heartbeat between both the racks must be successful.

3. Verify that the available roles of both instances have moved to STANDBY_ERROR on Rack-1.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

4. Verify that the available roles of both instances have moved to PRIMARY on Rack-2.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "PRIMARY"
```

5. Perform rolling upgrade (or) non-graceful upgrade using system mode shutdown/running as per the requirement on Rack-1. To allow replication to finish, give a 5-minute gap between the GR switchover and shutdown.

6. Perform the following steps post completion of the upgrade procedure. Perform health check on Rack-1 and ensure the PODs have come up and Rack-1 is healthy.

7. Verify that the available roles of both instances remain in STANDBY_ERROR mode on Rack-1.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

8. Initiate reset role for both instances on Rack-1 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

9. Verify that the roles of both instances have moved to STANDBY on Rack-1.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "STANDBY"
```

10. Initiate switch role for instance-id 1 on Rack-2 to STANDBY. This step transitions the available roles of Rack-2 from PRIMARY/PRIMARY to STANDBY_ERROR/PRIMARY and Rack-1 from STANDBY/STANDBY to PRIMARY/STANDBY.

```
geo switch-role instance-id 1 role standby [failback-interval 0]
```

11. Verify that the available roles of the instances on Rack-2 are in STANDBY_ERROR/PRIMARY.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "PRIMARY"
```

12. Verify that the available roles of both instances on Rack-1 are in PRIMARY/STANDBY.

```
show role instance-id 1
 result "PRIMARY"

show role instance-id 2
 result "STANDBY"
```

13. Initiate reset role for instance-id 1 on Rack-2 to STANDBY. This step transitions the roles of Rack-2 from STANDBY_ERROR/PRIMARY to STANDBY/PRIMARY.

```
geo reset-role instance-id 1 role standby
```

14. Verify that the available roles of both instances on Rack-2 are in STANDBY/PRIMARY.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
 result "PRIMARY"
```

Upgrading the Rack-2, when the GR is Enabled:

1. Verify that the available roles of both instances on Rack-2 are in STANDBY/PRIMARY.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "PRIMARY"
```

2. Initiate switch role for both instances on Rack-2 to STANDBY with failback-interval of 0 seconds. This step transitions the roles from STANDBY/PRIMARY to STANDBY_ERROR/STANDBY_ERROR.

```
geo switch-role instance-id 1 role standby [failback-interval 0]
geo switch-role instance-id 2 role standby [failback-interval 0]
```

3. Verify that the available roles of both instances move to STANDBY_ERROR on Rack-2.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

4. Verify that the available roles of both instances move to PRIMARY on Rack-1.

```
show role instance-id 1
 result "PRIMARY"

show role instance-id 2
 result "PRIMARY"
```

5. Perform rolling upgrade (or) non-graceful upgrade via system mode shutdown/running as per the requirement on Rack-2.
6. Perform the subsequent steps post completion of the upgrade procedure. Perform the health check on Rack-2 and ensure the PODs have come up and Rack-2 is healthy.
7. Verify that the available roles of both the instances remain in STANDBY_ERROR on Rack-2.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

8. Initiate reset role for both instances on Rack-2 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

9. Verify that the available roles of both instances move to STANDBY on Rack-2.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "STANDBY"
```

10. Initiate switch role for instance-id 2 on Rack-1 to STANDBY. This step transitions the available roles of Rack-1 from PRIMARY/PRIMARY to PRIMARY/STANDBY_ERROR and Rack-2 from STANDBY/STANDBY to STANDBY/PRIMARY.

    ```
    geo switch-role instance-id 2 role standby [failback-interval 0]
    ```

11. Verify that the available roles of both instances on Rack-1 are in PRIMARY/STANDBY_ERROR.

    ```
    show role instance-id 1
    result "PRIMARY"

    show role instance-id 2
    result "STANDBY_ERROR"
    ```

12. Verify that the available roles of both instances on Rack-2 are in STANDBY/PRIMARY.

    ```
    show role instance-id 1
    result "STANDBY"

    show role instance-id 2
    result "PRIMARY"
    ```

13. Initiate reset role for instance-id 2 on Rack-1 to STANDBY. This step transitions the roles of Rack-1 from PRIMARY/STANDBY_ERROR to PRIMARY/STANDBY.

    ```
    geo reset-role instance-id 2 role standby
    ```

14. Verify that the available roles of both the instances on Rack-1 are in PRIMARY/STANDBY.

    ```
    show role instance-id 1
    result "PRIMARY"

    show role instance-id 2
    result "STANDBY"
    ```

GR CLI

The following section provides information on GR CLI based commands.

Geo Switch Role

To switch the GR role, initiate the command on the primary rack (for example, role **PRIMARY** to **STANDBY** only), and use the following command.

geo switch-role { role *primary | standby* **instance-id** *gr_instanceId* **[failback-interval** *failback_interval* **] }**

NOTES:

- **role** *role*—Specify the new role for the given rack.

  The roles can be *primary* or *standby*. It's mandatory to trigger manual switchover from primary role for a specific GR instance ID.

- **instance-id** *gr_instanceId*—Specify the GR Instance ID

- **failback-interval** is an optional command to provide backward compatibility of upgrades between releases. The recommended value of **failback-interval** is 0.

Important **`geo switch-role`** command triggers manual failover from one rack to another rack for specific instance ID. The rack which triggers the failover changes from the **`PRIMARY`** role to the **`STANDBY_ERROR`** role. In between, the rack which triggers the failover, sends a failover (Trigger GR) message to another rack. The other rack which receives the failover message changes from the **`STANDBY`** role to the **`PRIMARY`** role.

Geo Reset Role

To reset the GR instance role (for example, roles from **`STANDBY_ERROR`** to **`STANDBY`** to **`PRIMARY`**), use the following sample commands:

`geo reset-role role` *`role`* **`instance-id`** *`gr_instanceId`*

NOTES:

- **role** *role*—Specify the new role for the given rack.

  The role can be **`PRIMARY`** or **`STANDBY`**.

- **instance-id** *gr_instanceId*—Specify the GR Instance ID.

Important `The command` **`geo reset-role`** triggers change in the role for the given instance on the local rack. The remote rack does not receive any message for the same command. It is only possible to change the role for the given instance ID from **`STANDBY_ERROR`** to **`STANDBY`** and **`STANDBY`** to **`PRIMARY`**. Another role change is not possible.

Troubleshooting

This section describes about various applicable troubleshooting scenarios.

show/clear Commands

This section describes show/clear commands that help in debugging issues.

clear subscriber

To clear gr-instance aware subscriber, use the following command:

`clear subscriber all gr-instance` *`gr_instanceId`*

Note **`gr-instance`** is optional parameter. If **gr-instance** is not specified, `show subscriber all` considers the local instance-id of that rack.

Example

The following is a configuration example.

```
clear subscriber all gr-instance 1
result
ClearSubscriber Request submitted
```

show BFD Status

To view the BFD status of neighbors, use the following command:

show bfd-neighbor

Example

The following is a list of few configuration examples:

```
show bfd-neighbor
status-details

-----example-bgp-ep-1 ----

Peer             Status

 209.165.202.142  STATE_DOWN
-----example-bgp-ep-2 ----

Peer             Status

 209.165.202.142  STATE_DOWN
```

```
show bfd-neigbor
status-details

-----bgpspeaker-pod-1 ----

Peer             Status

 209.165.202.131
-----bgpspeaker-pod-2 ----

Peer             Status

 209.165.202.131   STATE_UP
```

show BGP Global

To view BGP global configuration, use the following command:

show bgp-global

Example

The following is a list of few configuration examples:

```
show bgp-global
global-details
-----example-bgp-ep-2 ----
AS:        65000
Router-ID: 209.165.202.149
Listening Port: 179, Addresses: 209.165.202.149
-----example-bgp-ep-1 ----
AS:        65000
Router-ID: 209.165.202.148
Listening Port: 179, Addresses: 209.165.202.148
```

```
show bgp-global
global-details

-----bgpspeaker-pod-2 ----
AS:        65061
Router-ID: 209.165.202.132
Listening Port: 179, Addresses: 209.165.202.132
```

show bgp kernel route

To view BGP kernel configured routes, use the following command:

show bgp-kernel-route kernel-route

Example

The following is a list of few configuration examples:

```
show bgp-kernel-route
kernel-route

-----example-bgp-ep-2 ----

 DestinationIP  SourceIP        Gateway

-----example-bgp-ep-1 ----

 DestinationIP      SourceIP          Gateway
 209.165.202.133    209.165.202.148   209.165.202.142
 209.165.202.134    209.165.202.148   209.165.202.142
```

```
show bgp-kernel-route
kernel-route

-----bgpspeaker-pod-2 ----

 DestinationIP        SourceIP            Gateway

 209.165.202.135    209.165.202.132    209.165.202.131

-----bgpspeaker-pod-1 ----

 DestinationIP  SourceIP        Gateway
```

show bgp neighbors

To view BGP neighbors status, use the following command

show bgp-neighbors neighbor-details
show bgp-neighbors ip *ip_address* **neighbor-details**

Example

The following is a list of few configuration examples:

```
show bgp-neighbors neighbor-details
-----example-bgp-ep-1 ----
Peer             AS  Up/Down State       |#Received  Accepted
209.165.202.142 60000 00:25:06 Establ      |        3         3
-----example-bgp-ep-2 ----
Peer             AS Up/Down State       |#Received  Accepted
209.165.202.142 60000   never Idle        |        0         0
```

```
show bgp-neighbors ip 209.165.202.142 neighbor-details
-----example-bgp-ep-2 ----
BGP neighbor is 209.165.202.142, remote AS 60000
  BGP version 4, remote router ID unknown
  BGP state = ACTIVE
  BGP OutQ = 0, Flops = 0
  Hold time is 0, keepalive interval is 0 seconds
  Configured hold time is 90, keepalive interval is 30 seconds

  Neighbor capabilities:
    multiprotocol:
        ipv4-unicast:   advertised
    route-refresh:      advertised
    extended-nexthop:   advertised
        Local:  nlri: ipv4-unicast, nexthop: ipv6
    4-octet-as: advertised
  Message statistics:
                        Sent       Rcvd
    Opens:               130          0
    Notifications:         0          0
    Updates:               0          0
    Keepalives:            0          0
    Route Refresh:         0          0
    Discarded:             0          0
    Total:               130          0
  Route statistics:
    Advertised:            0
    Received:              0
    Accepted:              0

-----example-bgp-ep-1 ----
BGP neighbor is 209.165.202.142, remote AS 60000
  BGP version 4, remote router ID 209.165.202.136
  BGP state = ESTABLISHED, up for 00:25:20
  BGP OutQ = 0, Flops = 0
  Hold time is 90, keepalive interval is 30 seconds
  Configured hold time is 90, keepalive interval is 30 seconds

  Neighbor capabilities:
    multiprotocol:
        ipv4-unicast:   advertised and received
    route-refresh:      advertised and received
    extended-nexthop:   advertised
        Local:  nlri: ipv4-unicast, nexthop: ipv6
    4-octet-as: advertised and received
  Message statistics:
                        Sent       Rcvd
    Opens:                 1          1
    Notifications:         0          0
    Updates:               1          1
    Keepalives:           51         51
    Route Refresh:         0          0
    Discarded:             0          0
    Total:                53         53
  Route statistics:
    Advertised:            0
    Received:              3
    Accepted:              3
```

show bgp route summary

To view BGP route summary, use the following command:

show bgp-route-summary

Example

The following is a configuration example.

```
show bgp-route-summary
route-details
-----example-bgp-ep-1 ----
Table afi:AFI_IP safi:SAFI_UNICAST
Destination: 5, Path: 5
-----example-bgp-ep-2 ----
Table afi:AFI_IP safi:SAFI_UNICAST
Destination: 2, Path: 2
```

show BGP Routes

To view BGP routes information, use the following command:

show bgp-routes

Example

The following is a configuration example:

```
show bgp-routes
bgp-route

-----example-bgp-ep-1 ----
   Network              Next Hop             AS_PATH      Age        Attrs
*> 209.165.202.133/24   209.165.202.142        60000      00:25:55   [{Origin: i} {Med: 0}]
*> 209.165.200.225/32   209.165.202.148                  00:26:00   [{Origin: e} {LocalPref:
 100} {Med: 600}]
*> 209.165.202.134/24   209.165.202.142        60000      00:25:55   [{Origin: i} {Med: 0}]
*> 209.165.202.140/24   209.165.202.142        60000      00:25:55   [{Origin: i} {Med: 0}]
*> 209.165.202.146/32   209.165.202.148                  00:26:00   [{Origin: e} {LocalPref:
 100} {Med: 600}]

-----example-bgp-ep-2 ----
   Network              Next Hop             AS_PATH      Age        Attrs
*> 209.165.200.225/32   209.165.202.149                  00:26:24   [{Origin: e} {LocalPref:
 100} {Med: 600}]
*> 209.165.202.146/32   209.165.202.149                  00:26:24   [{Origin: e} {LocalPref:
 100} {Med: 600}]
```

show endpoint

To view endpoints that are now gr-instance aware, use the following command:

show endpoint all grInstance *gr_instanceId*

Note **grInstance** is optional parameter. If **grInstance** is not specified, `show subscriber all` considers the local instance-id of that rack.

Example

The following is a configuration example:

```
show endpoint all grInstance 1
                                                                                START

STOPPED  GR
```

```
ENDPOINT                          ADDRESS        TYPE  STATUS   INTERFACE  INTERNAL  TIME
     TIME     INSTANCE
-------------------------------------------------------------------------------------------------
209.165.202.137:2123        209.165.202.137:2123  Udp   Started             false     10
hours  <none>   1
Gtpu:209.165.202.137:2152   209.165.202.137:2152  Udp   Started  GTPU       false     10
hours  <none>   1
N4:209.165.202.137:8806     209.165.202.137:8806  Udp   Started  N4         false     10
hours  <none>   1
S2B-GTP                     209.165.202.138:2124  Udp   Started  s2b        false     10
hours  <none>   1
S5-GTP                      209.165.202.138:2125  Udp   Started  s5         false     10
hours  <none>   1
S5S8S2B-GTP                 209.165.202.138:2123  Udp   Started  s5s8s2b    false     10
hours  <none>   1
Sxa:209.165.202.137:8805    209.165.202.137:8805  Udp   Started  SXA        false     10
hours  <none>   1
n10-1                       209.165.202.139:9010  Rest  Started  N10-1      false     10
hours  <none>   1
n11-1                       209.165.202.139:9011  Rest  Started  N11-1      false     10
hours  <none>   1
n40-1                       209.165.202.139:9040  Rest  Started  N40-1      false     10
hours  <none>   1
n7-1                        209.165.202.139:9007  Rest  Started  N7-1       false     10
hours  <none>   1
sbi-1                       209.165.202.139:8090  Rest  Started  SBI-1      false     10
hours  <none>   1
```

show ETCD/Cache Pod Replication

To view replication details for etcd and cache-pod data, use the following command:

show georeplication checksum instance-id *gr_instanceId*

Example

The following is a configuration example:

```
show georeplication checksum instance-id
Value for 'instance-id' (<string>): 1
checksum-details
--              ----    --------
 ID             Type    Checksum
 --             ----    --------
 1              ETCD    1617984439
 IPAM           CACHE   1617984439
 NRFCache       CACHE   1617984439
 NRFSubs        CACHE   1617984439
 IDMGR          CACHE   1617984439
 NRFMgmt        CACHE   1617984439
```

show role

To view the current role of the GR instance, use the following command:

show role instance-id *gr_instanceId*

Note The following is a list of possible values for the role:

- **PRIMARY**
- **STANDBY**
- **FAILOVER_INIT**
- **FAILOVER_COMPLETE**
- **STANDBY_ERROR**
- **FAILBACK_STARTED**

Example

The following is a list of few configuration examples:

```
show role instance-id 1
result
"PRIMARY"

show role instance-id 2
result
"STANDBY"
```

show ipam dp with type and address

To view the instance ID and flag to indicate chunk for remote instance, use the following command:

show ipam dp { *dp_type* } { *addr_type* }

NOTES:

- **dp** *dp_type*—Specify DP type.
- *addr_type*—Specify IPv4/IPv6 address type.

Example

The following is a configuration example.

```
show ipam dp 209.165.202.145:209.165.202.144  ipv4-addr
=====================================================================
Flag Indication: S(Static) O(Offline) R(For Remote Instance)
G:N/P Indication: G(GR InstId) N(Native NM InstId) P(Peer NM InstId)
=====================================================================
StartAddress    EndAddress      AllocContext                    Route              G:N/P
Utilization Flag
=====================================================================
209.165.200.240 209.165.200.243 209.165.202.145:209.165.202.144 209.165.200.240/24 1:0/1
0.00%        R
=====================================================================
```

show ipam dp

To view all the instances this DP has chunks from, use the following command:

show ipam dp *dp_name*

NOTES:

- **dp** *dp_name*—Specify data plane allocation name.

Example

The following is a configuration example.

```
show ipam dp 209.165.202.145:209.165.202.144
-------------------------------------------------------
Ipv4Addr [Total/Used/Utilization] = 257 / 1 / 0.39%
Ipv6Addr [Total/Used/Utilization] = 0 / 0 / 0.00%
Ipv6Prefix [Total/Used/Utilization] = 2048 / 0 / 0.00%
Instance ID = 1
-------------------------------------------------------
```

show ipam pool

To view instance ID information under which pool is configured, use the following command:

show ipam pool *pool_name*

NOTES:

- **pool** *pool_name*—Specify pool name.

Example

The following is a list of few configuration examples.

```
show ipam pool

=============================================================================================
PoolName                      Ipv4Utilization  Ipv6AddrUtilization  Ipv6PrefixUtilization
=============================================================================================
poolv6DNN2                    0.00%            0.00%                0.00%
poolv6                        0.00%            0.00%                0.00%
poolv4vDNN                    0.00%            0.00%                0.00%
poolv4DNN2                    0.00%            0.00%                0.00%
poolv4                        0.00%            0.00%                0.00%
poolv6vDNN                    0.00%            0.00%                0.00%
poolv4DNN3                    -                -                    -
=============================================================================================

show ipam pool poolv4DNN3
-------------------------------------------------------
Ipv4Addr [Total/Used/Utilization] = 2814 / 0 / -
Ipv6Addr [Total/Used/Utilization] = 0 / 0 / -
Ipv6Prefix [Total/Used/Utilization] = 65536 / 0 / -
Instance ID = 1
isStatic = true
-------------------------------------------------------

show ipam pool poolv4
-------------------------------------------------------
 Ipv4Addr   [Total/Used/Utilization] = 2814 / 0 / 0.00%
 Ipv6Addr   [Total/Used/Utilization] = 0 / 0 / 0.00%
 Ipv6Prefix [Total/Used/Utilization] = 0 / 0 / 0.00%
 Instance ID                         = 1
-------------------------------------------------------
```

show nrf discovery-info discovery-filter

To view GR Instance ID information to determine for which GR instance the discovery filter information belongs, use the following command:

show nrf discovery-info *nf_type* **discovery-filter**

Example

The following is a configuration example.

```
=====================================================
-----------------------------------------------------
Discovery Filter: dnn=intershat;
Expiry Time: 1580146356
GR Instance ID: 1
-----------------------------------------------------
=====================================================================
```

show nrf discovery-info

To view GR Instance ID information to determine for which GR instance the discovery information belongs, use the following command:

show nrf discovery-info

Example

The following is a configuration example.

```
show nrf discovery-info
=====================================================
------Discovered NFs:-------
    NF Type: AMF
    Number of Discovery Filters: 15
    Number of NF Profiles: 15
    GR Instance ID: 1
------Discovered NFs:-------
    NF Type: UDM
    Number of Discovery Filters: 1
    Number of NF Profiles: 3
    GR Instance ID: 2
=====================================================
```

show nrf registration-info

To view GR Instance ID information to determine which GR instance the registration information belongs to, use the following command:

show nrf registration-info

Example

The following is a configuration example.

```
show nrf registration-info
=====================================================================
 NF Status: Not Registered
 Registration Time:
 Active MgmtEP Name:
 Heartbeat Duration: 0
```

```
 GR Instance ID: 1
========

show nrf registration-info

======================================================================
 Gr-instance:
 NF Status: Not Registered
 Registration Time:
 Active MgmtEP Name:
 Heartbeat Duration: 0
 Uri:
 Host Type:

======================================================================
 Gr-instance:
 NF Status: Not Registered
 Registration Time:
 Active MgmtEP Name:
 Heartbeat Duration: 0
 Uri:
 Host Type:

======================================================================
```

show nrf subscription-info

To view GR Instance ID information to determine for which GR instance the subscription information belongs, use the following command:

show nrf subscription-info

Example

The following is a configuration example.

```
show nrf subscription-info
======================================================================
NF Instance Id: f9882966-a253-32d1-8b82-c785b34a7cc9
SubscriptionID : subs123459
Actual Validity Time : 2020-01-21 12:39:45 +0000 UTC
Requested Validity Time : 2020-01-21 12:39:45 +0000 UTC
GR Instance ID: 1
======================================================================
```

show peers

To view peers that are now gr-instance aware, use the following command:

show peers all grInstance *gr_instanceId*

Note **grInstance** is optional parameter. If **grInstance** is not specified, `show subscriber all` considers the local instance-id of that rack.

Example

The following is a configuration example.

```
show peers all grInstance 1
                                                                  POD              CONNECTED
ADDITIONAL  INTERFACE  GR
ENDPOINT  LOCAL ADDRESS   PEER ADDRESS     DIRECTION  INSTANCE TYPE TIME     RPC DETAILS
NAME       INSTANCE
-------------------------------------------------------------------------------------------------------
<none>  209.165.202.139 209.165.201.22:8001 Outbound rest-ep-0 Rest 10 hours UDM <none>
n10        1
<none>  209.165.202.139 209.165.201.22:8002 Outbound rest-ep-0 Rest 10 hours AMF <none>
n11        1
<none>  209.165.202.139 209.165.201.22:8003 Outbound rest-ep-0 Rest 10 hours PCF <none>
n7         1
<none>  209.165.202.139 209.165.201.22:8004 Outbound rest-ep-0 Rest 10 hours CHF <none>
n40        1
<none>  209.165.202.139 209.165.201.22:9040 Outbound rest-ep-0 Rest 10 hours CHF <none>
n40
```

show role

To view the instance role, use the following command:

show role

Example

The following is a list of few configuration examples:

```
show role instance-id 2
result "PRIMARY"
```

```
show role instance-id 1
result "PRIMARY"
```

show subscriber

To view subscriber details that are made gr-instance aware, use the following command:

show subscriber { all | gr-instance *gr_instanceId* **}**

Note `show subscriber all` displays only the local instance subscriber details.

gr-instance is optional parameter. If **gr-instance** is not specified, `show subscriber all` considers the local instance-id of that rack.

Example

The following is a configuration example.

```
show subscriber gr-instance 1 all
subscriber-details
{
"subResponses": [
[
""
],
[
""
],
[
"roaming-status:homer",
```

```
"supi:imsi-123456789300001",
"gpsi:msisdn-22331010301010",
"psid:1",
"dnn:intershat",
"emergency:false",
"rat:nr",
"access:3gpp access",
"connectivity:5g",
"udm-uecm:209.165.202.150",
"udm-sdm:209.165.202.150",
"auth-status:unauthenticated",
"pcfGroupId:PCF-*",
"policy:2",
"pcf:209.165.202.152",
"upf:209.165.202.154",
"upfEpKey:209.165.202.154:209.165.202.158",
"ipv4-addr:v4pool1/209.165.200.250",
"ipv4-pool:v4pool1",
"ipv4-range:v4pool1/209.165.200.249",
"ipv4-startrange:v4pool1/209.165.200.250",
"id-index:1:0:0:32768",
"id-value:8",
"chfGroupId:CHF-*",
"chf:209.165.202.151",
"amf:209.165.202.153",
"peerGtpuEpKey:209.165.202.154:209.165.202.155",
"namespace:smf",
"nf-service:smf"
]
]
}
```

Monitor Subscriber

To capture messages for subscriber (gr-instance aware), use the following command:

Note In 2021.02 and later releases, the **namespace** keyword is deprecated and replaced with the **nf-service** keyword.

NOTES:

Example

The following is a configuration example.

```
monitor subscriber imsi 123456789 gr-instance 1
supi: imsi-123456789
captureDuration: 300
enableInternalMsg: false
enableTxnLog: false
namespace(deprecated. Use nf-service instead.): none
nf-service: none
gr-instance: 1
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   295  100    98  100   197  10888  21888 --:--:-- --:--:-- --:--:-- 29500
Command: --header Content-type:application/json --request POST --data
{"commandname":"mon_sub","parameters":{"supi":"imsi-123456789","duration":300,
"enableTxnLog":false,"enableInternalMsg":false,"action":"start","namespace":"none",
"nf-service":"none","grInstance":1}} http://oam-pod:8879/commands
Result start mon_sub, fileName
```

```
->logs/monsublogs/none.imsi-123456789_TS_2021-04-09T09:59:59.964148895.txt
Starting to tail the monsub messages from file:
logs/monsublogs/none.imsi-123456789_TS_2021-04-09T09:59:59.964148895.txt
Defaulting container name to oam-pod.
Use 'kubectl describe pod/oam-pod-0 -n smf' to see all the containers in this pod.
```

Monitor Protocol

To capture packets on different interfaces (gr-instance aware), use the following command:

monitor protocol { interface *interface_name* **[capture-duration** *duration* **| gr-instance***gr_instance* **| pcacp yes | |] | list [|] }**

Warning Use the **monitor protocol interface** command cautiously, as it may disrupt user services. When used in a live network, it can cause session delays, call drops, and degrade system performance.

Note

- **interface** *interface_name*—Specify the interface name on which PCAP is captured. This CLI allows the configuration of multiple interface names in a single CLI command.
- **capture-duration** *duration*—Specify the duration in seconds during which pcap is captured. The default is 300 seconds (5 minutes).
- The configured interface names can be retrieved using the **show endpoint** CLI command.
- **pcap yes**—Configure this option to enable PCAP file generation. By default, this option is disabled.
- **list**—Monitor protocol list files.

Example

The following is a configuration example.

```
monitor protocol interface sbi gr-instance 1
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   220  100    95  100   125   8636  11363 --:--:-- --:--:-- --:--:-- 20000
Command: --header Content-type:application/json --request POST --data
{"commandname":"mon_pro","parameters":{"interface":"sbi","duration":300,"action":
"start","enable_pcap":false,"grInstance":1}} http://oam-pod:8879/commands
Result start mon_pro, fileName
->logs/monprologs/sessintfname_sbi_at_2021-04-30T05:26:22.712229347.txt
Starting to tail the monpro messages from file:
logs/monprologs/sessintfname_sbi_at_2021-04-30T05:26:22.712229347.txt
Defaulting container name to oam-pod.
Use 'kubectl describe pod/oam-pod-0 -n cn' to see all of the containers in this pod.
```

Geographic Redundancy OAM Support

This section describes operations, administration, and maintenance information for this feature.

Prerequisites for RMA Process

For GR deployment, the node-monitor pods starts automatically. During RMA procedure, the node-monitor pod automatically shutdown the rack if multi-compute failure is detected when the node is drain and deleted.

For more information on RMA (Return Merchandise Authorization), see *SMI Cluster RMA* section in the *Ultra Cloud Core Subscriber Microservices Infrastructure - Operations Guide*.

Before starting RMA process, perform the following:

1. Switch the role for both the instance to other rack using `geo switch-role role` command and make sure the target rack for RMA is in STANDBY_ERROR role for both the instances.
2. Disable the node-monitor pod.
   a. Take the backup of daemonsets.

      ```
      kubectl get daemonsets node-monitor -n cn -o yaml > node-monitor.yaml
      ```

   b. Delete node-monitor pods.

      ```
      kubectl delete daemonsets node-monitor -n cn
      ```

3. Continue with RMA procedure. For more information, see the link.
4. Once RMA procedure is complete, check if the node-monitor pods are already spawned.

   ```
   kubectl get pods -n cn -o wide | grep node-monitor
   ```

   If the node-monitor pods have not started, restart them.

   ```
   kubectl create -f node-monitor.yaml
   ```

Note `node-monitor.yaml` file is same as in Step 2.a, on page 580.

5. Correct the role for the instances accordingly.

Note For both earlier and current SMI versions:

- If you are replacing hardware components during an RMA procedure that contain firmware, such as an mLOM card, before adding the repaired or replaced node back to the cluster, you must run the HUU (Host Upgrade Utility) to ensure that the component is compatible with the system before syncing the node back into service.
- As part of RMA, if you remove a node from the cluster and before you return it to the manufacturer, you must purge all data on the device as per instructions provided by the hardware vendor.

Health Check

The following section provides information on GR setup health check.

- All critical pods are in good condition to serve user traffic.

Use the following command to check whether GR and CDL related pods are in Running state.

```
kubectl get pods -n cn-cn1 -o wide | grep georeplication-pod
kubectl get pods -n cn-cn1 -o wide | grep cdl
kubectl get pods -n cn-cn1 -o wide | grep mirror-maker
```

- Keepalived pods are in healthy state to monitor all VIPs which are configured for check-interface/check-port.

  Use the following command to check whether keepalived pods in “smi-vips” namespace are in “Running” state.

  ```
  kubectl get pods -n smi-vips
  ```

- Health-check of pods related to CDL: Check the status of CDL db-endpoint, slot and indexes. All should be in STARTED or ONLINE state for both System IDs 1 and 2.

  ```
  cdl show status
  message params: {cmd:status mode:cli dbName:session sessionIn:{mapId:0 limit:500 key:
  purgeOnEval:0 filters:[] nextEvalTsStart:0 nextEvalTsEnd:0 allReplicas:false
  maxDataSize:4096} sliceName:}
  db-endpoint {
      endpoint-site {
          system-id 1
          state STARTED
          total-sessions 4
          site-session-count 2
          total-reconciliation 0
          remote-connection-time 66h37m31.36054781s
          remote-connection-last-failure-time 2021-07-13 11:24:10.233825924 +0000 UTC
          slot-geo-replication-delay 2.025396ms
      }
      endpoint-site {
          system-id 2
          state STARTED
          total-sessions 4
          site-session-count 2
          total-reconciliation 0
          remote-connection-time 66h58m49.83449066s
          remote-connection-last-failure-time 2021-07-13 11:02:51.759971655 +0000 UTC
          slot-geo-replication-delay 1.561816ms
      }
  }
  slot {
      map {
          map-id 1
          instance {
              system-id 1
              instance-id 1
              records 4
              capacity 2500000
              state ONLINE
              avg-record-size-bytes 1
              up-time 89h38m37.335813523s
              sync-duration 9.298061ms
          }
          instance {
              system-id 1
              instance-id 2
              records 4
              capacity 2500000
              state ONLINE
              avg-record-size-bytes 1
              up-time 89h39m11.1268024s
  ```

```
                sync-duration 8.852556ms
            }
            instance {
                system-id 2
                instance-id 1
                records 4
                capacity 2500000
                state ONLINE
                avg-record-size-bytes 1
                up-time 89h28m38.274713022s
                sync-duration 8.37766ms
            }
            instance {
                system-id 2
                instance-id 2
                records 4
                capacity 2500000
                state ONLINE
                avg-record-size-bytes 1
                up-time 89h29m37.934345015s
                sync-duration 8.877442ms
            }
        }
    }
    index {
        map {
            map-id 1
            instance {
                system-id 1
                instance-id 1
                records 4
                capacity 60000000
                state ONLINE
                up-time 89h38m16.119032086s
                sync-duration 2.012281769s
                leader false
                geo-replication-delay 10.529821ms
            }
            instance {
                system-id 1
                instance-id 2
                records 4
                capacity 60000000
                state ONLINE
                up-time 89h39m8.47664588s
                sync-duration 2.011171261s
                leader true
                leader-time 89h38m53.761213379s
                geo-replication-delay 10.252683ms
            }
            instance {
                system-id 2
                instance-id 1
                records 4
                capacity 60000000
                state ONLINE
                up-time 89h28m29.5479133s
                sync-duration 2.012101957s
                leader false
                geo-replication-delay 15.974538ms
            }
            instance {
                system-id 2
                instance-id 2
```

```
                records 4
                capacity 60000000
                state ONLINE
                up-time 89h29m11.633496562s
                sync-duration 2.011566639s
                leader true
                leader-time 89h28m51.29928233s
                geo-replication-delay 16.213323ms
            }
        }
    }
```

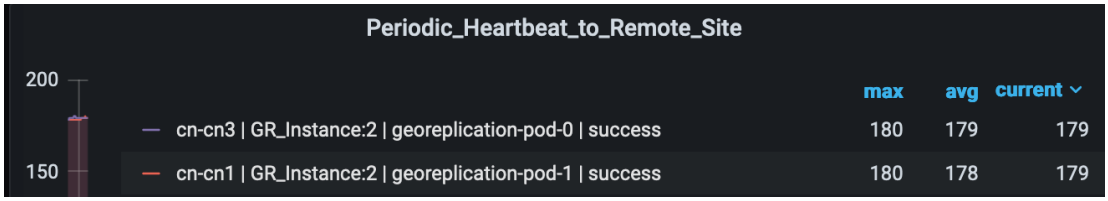
- CDL replication status

  Check whether four gRPC connections are established between the CDL EP session pods (of each namespace) across the racks in **GRPC_Connections_to_RemoteSite** panel of **CDL Replication Stats** Grafana dashboard. Check Grafana on both racks.



- Admin port status between the racks for geo-replication.

  Check heartbeat messages between geo-replication pods across the racks in **Periodic_Heartbeat_to_Remote_Site** panel of **GR Statistics** Grafana dashboard.



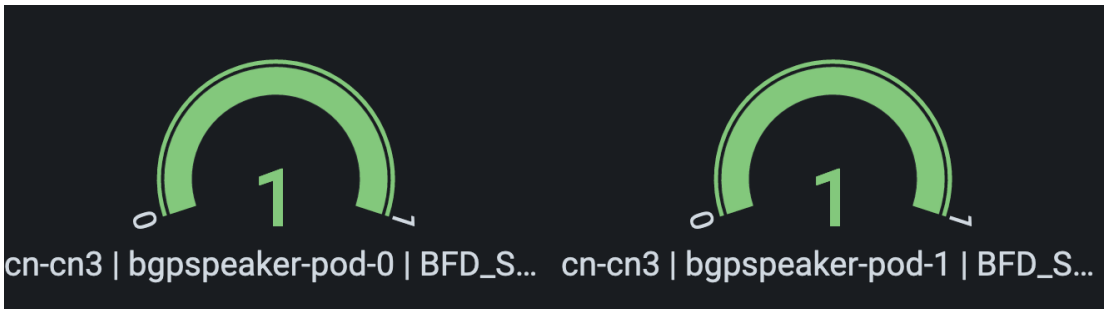
- BGP/BFD link status on rack

  Check whether neighborship with BGP peers is established in **BGP Peers** panel of **BGP, BFD Statistics** Grafan dashboard.

BGP Peers

Time	as_path	namespace	peer_ip	pod	Value
2021-07-16 06:29:18	3333	cn-cn1	192.204.10.6	bgpspeaker-pod-0	1
2021-07-16 06:29:18	3333	cn-cn1	192.204.10.6	bgpspeaker-pod-1	1
2021-07-16 06:29:18	3333	cn-cn3	192.204.18.6	bgpspeaker-pod-0	1
2021-07-16 06:29:18	3333	cn-cn3	192.204.18.6	bgpspeaker-pod-1	1

Check whether BFD link is in connected state in **BFD Link Status** panel of **BGP, BFD Statistics** Grafana dashboard.



- Roles of each instances are in healthy state

  Check that in each rack the roles are not in STANDBY_ERROR state at any point of time.

  - **Active/Standby Model**: Roles should be in the following states on each rack

    Rack-1:

    ```
    show role instance-id 1
    result "PRIMARY"
    show role instance-id 2
    result "PRIMARY"
    ```

    Rack-2:

    ```
    show role instance-id 1
    result "STANDBY"
    show role instance-id 2
    result "STANDBY"
    ```

  - **Active/Active Model**: Roles should be in the following states on each rack.

    Rack-1:

    ```
    show role instance-id 1
    result "PRIMARY"
    show role instance-id 2
    result "STANDBY"
    ```

    Rack-2:

    ```
    show role instance-id 1
    result "STANDBY"
    show role instance-id 2
    result "PRIMARY"
    ```

Recovery Procedure

On Rack-1

1. Verify that roles of both instances on Rack-1 are in STANDBY_ERROR.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

2. Initiate reset role for both instances on Rack-1 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

3. Verify that roles of both instances have moved to STANDBY on Rack-1.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "STANDBY"
```

4. Initiate switch role for instance-id 1 on Rack-2 to STANDBY with failback-interval of 30 seconds. This step transitions the roles of Rack-2 from PRIMARY/PRIMARY to STANDBY_ERROR/PRIMARY and Rack-1 from STANDBY/STANDBY to PRIMARY/STANDBY.

```
geo switch-role instance-id 1 role standby [failback-interval 0]
```

5. Verify that roles of both instances on Rack-2 are in STANDBY_ERROR/PRIMARY.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "PRIMARY"
```

6. Verify that roles of both instances on Rack-1 are in PRIMARY/STANDBY.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "STANDBY"
```

7. Initiate reset role for instance-id 1 on Rack-2 to STANDBY. This step transitions the roles of Rack-2 from STANDBY_ERROR/PRIMARY to STANDBY/PRIMARY.

```
geo reset-role instance-id 1 role standby
```

8. Verify that the roles of Rack-2 are in STANDBY/PRIMARY.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "PRIMARY"
```

On Rack-2

1. Verify that roles of both the instances on Rack-2 are in STANDBY_ERROR.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

2. Initiate reset role for both instances on Rack-2 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

3. Verify that the roles of both the instances move to STANDBY on Rack-2.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "STANDBY"
```

4. Initiate switch role for instance-id 2 on Rack-1 to STANDBY. This step transitions roles of Rack-1 from PRIMARY/PRIMARY to PRIMARY/STANDBY_ERROR and Rack-2 from STANDBY/STANDBY to STANDBY/PRIMARY.

```
geo switch-role instance-id 2 role standby [failback-interval 0]
```

5. Verify that roles of instances on Rack-1 are in PRIMARY/STANDBY_ERROR mode.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "STANDBY_ERROR"
```

6. Verify that roles of instances on Rack-2 are in STANDBY/PRIMARY mode.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "PRIMARY"
```

7. Initiate reset role for instance-id 2 on Rack-1 to STANDBY. This step transitions the roles of Rack-1 from PRIMARY/STANDBY_ERROR to PRIMARY/STANDBY.

```
geo reset-role instance-id 2 role standby
```

8. Verify that roles of instances on Rack-1 are in PRIMARY/STANDBY.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "STANDBY"
```

Key Performance Indicators (KPIs)

The following section describes KPIs.

ETCD/Cachepod Replication KPIs

The following table lists ETCD/Cachepod Replication KPIs.

Table 203: geo_replication_total KPIs

KPI Name	Description	Labels	Possible Values
geo_replication _total	This KPI displays total number of replication requests/responses for various Sync types and Replication types.	ReplicationRequest Type	Request / Response
		ReplicationSync Type	Immediate / Deferred / Pull
		ReplicationNode	ETCD / CACHE_POD / PEER
		ReplicationReceiver	Local / Remote
		status	True / False
		status_code	Error code/description

Geo Rejected Role Change KPIs

The following table lists Geo Rejected Role Change KPIs.

Table 204: Geo Rejected Role Change KPIs

KPI Name	Description	Labels	Possible Values
geo_RejectedRole Changed_total	This KPI displays the total number of rejected requests/calls received for STANDBY instance. After the count, the same instance is moved to PRIMARY.	RejectedCount	Number value indicating rejected calls/requests received for standby instance.
		GRInstance Number	1 / 2

Monitoring KPIs

The following table lists monitoring KPIs.

Table 205: geo_monitoring_total KPIs

KPI Name	Description	Labels	Possible Values
geo_monitoring _total	This KPI displays the total number of successful / failure messages of different kinds such as, heartbeat / remoteNotify / TriggerGR and so on.	ControlAction Type	AdminMonitoring ActionType / AdminRemote MessageAction Type / AdminRole ChangeActionType
		ControlAction NameType	MonitorPod / MonitorBfd / RemoteMsgHeartbeat / RemoteMsgNotifyFailover / RemoteMsgNotify PrepareFailover / RemoteMsgGetSiteStatus / RemoteClusterPodFailure / RemoteSiteRole Monitoring / TriggerGRApi / ResetRoleApi
		Admin Node	Any string value. For example, GR Instance ID or instance key / pod name
		Status Code	0 / 1001 / 1002 / 1003 / 1004 / 1005 / 1006 / 1007 / 1008 / received error code (1206, 1219, 2404, …)
		Status Message	

KPI Name	Description	Labels	Possible Values
			Success (0) / STANDBY_ERROR => STANDBY/STANDBY => PRIMARY (0) / Pod Failure (0) / CLI (0) / BFD Failure (0) / Decode Failure (1001) / remote status unavailable (1002) / target role does not support (1002) / Pod Failure (1002) / CLI (1002) / BFD Failure (1002) / site is down (1003) / Pod Failure (1003) / CLI (1003) / BFD Failure (1003) / Traffic Hit (1004) / Pod Failure (1004) / CLI (1004) / BFD Failure (1004) / current role is not STANDBY_ERROR/ STANDBY to reset role (1005) / resetRole: Key not found in etcd (1006) / monitoring threshold per pod is breached (1007) / Retry on heartbeat failure (1008) / received error message (No remote host available for this request / Selected remote host <remotehostname> has no client connection / Sla is expired for transaction / …)

Table 206: geo_replication_finalpull_total KPIs

KPI Name	Description	Label Names	Possible Values
geo_replication _finalpull_total	This KPI displays the total number of geo replications present in the final pull of the feature messages.	**MessageType**	It's a request or a response message type.
		TotalTimeTaken	It's the total time taken to process the request.
		GRInstanceNumber	It's the GR Instance ID in number from the list of the following: • 1 • 2 • Instance.1 • Instance.2

BFD KPIs

The following table lists BFD KPIs.

Table 207: BFD KPIs - 1

KPI Name	Description	Labels	Possible Values
bgp_speaker _bfd_status	This KPI displays BFD link status on BGP Speaker.	status	STATE_UP / STATE_DOWN
geo_bfd_ status	This KPI displays BFD link status on Geo POD.	status	STATE_UP / STATE_DOWN

Table 208: BFD KPIs - 2

KPI Name	Description	Gauge
bgp_speaker _bfd_status	This KPI displays BFD link status on BGP Speaker.	1 (UP) or 0 (DOWN)
geo_bfd_ status	This KPI displays BFD link status on Geo POD.	1 (UP) or 0 (DOWN)

Cross-rack-routing BFD Interface Monitoring

Table 209: Cross-rack-routing BFD Interface Monitoring KPIs

KPI Name	Description	Labels	Possible Values
geo_monitoring_ total	This KPI displays the total number of Gateway Down or LocalBFDInterface down messages when peer rack is down with the details of gateway IP or interface name.	ControlAction Type	AdminMonitoring ActionType
		ControlAction NameType	MonitorGateway / MonitorLocalBfdInterface
		AdminNode	gateway_ip / interface_name
		status	gateway ip is down from all proto node / local bfd interface is down from all proto node
		status_code	1012 / 1013
bgp_bfd_Monitor_ Interface_ status (Type - Gauge)	This KPI indicates each peer connection status. This connection is BFD interface configured and peers on the remote rack.	interface	<Local Rack Interface Name>
		peer_address	<Remote Rack neighbor Ip address>
		type	Bfd-Peer
bgp_bfd_Monitor_ Remote_Rack_ status (Type - Gauge)	This KPI indicates the status of remote rack. Current rack interface and remote rack peers are configured in as a part of BFD peering. Rack status is up if any of the connection from both the proto node is up. If connection is down at both the proto nodes, then this KPI indicates the remote rack status is down.	status	BFD_Remote_ Rack_STATUS

Local Interface Monitoring

Table 210: Local Interface Monitoring KPI

KPI Name	Description	Labels	Possible Values
geo_monitoring_ total	This KPI displays the total number of local interface down cases with the details of interface name.	ControlAction Type	AdminMonitoring ActionType
		ControlAction NameType	MonitorInterface
		AdminNode	interface_name
		status	Local interface is down from all proto node
		status_code	1014

GR Instance Information

Table 211: GR Instance Information KPI

KPI Name	Description	Labels	Possible Values
gr_instance_ information (Type – Guage)	This KPI displays the current role of the GR instance in the application.	gr_instance_id	Configured GR instances value (numerical value)

Geo Maintenance Mode

Table 212: Geo Maintenance Mode KPI

KPI Name	Description	Labels	Possible Values
geo_MaintenanceMode_ info (Type – Guage)	This KPI displays the current state of maintenance mode for the rack.	MaintenanceMode	0: false 1: true

Bulk Statistics

The following section provides details on GR-specific bulkstats.

```
bulk-stats query GR-BGP-Incoming-Failed-Routes
 expression "sum(bgp_incoming_failedrouterequest_total) by (namespace, interface, service_IP,
 next_hop, instance_id)"
 labels     [ instance_id interface next_hop service_IP ]
 alias      gr-bgp-routes-in
exit
bulk-stats query GR-Geo-Monitoring-Failure
 expression "sum(geo_monitoring_total{ControlActionNameType=~'MonitorPod|RemoteMsgHeartbeat|
```

```
RemoteMsgGetSiteStatus|RemoteSiteRoleMonitoring|RemoteClusterPodFailure|RemoteMsgNotifyFailover|
RemoteMsgNotifyPrepareFailover|MonitorVip',status!~'success|monitoring.*'}) by (namespace,

AdminNode, ControlActionType, ControlActionNameType,  pod, status, status_code)"
 labels     [ pod AdminNode ControlActionNameType status status_code ]
 alias      gr-geo-monitoring-failure
exit
bulk-stats query GR-Geo-Monitoring-Success
 expression "sum(geo_monitoring_total{ControlActionNameType=~'MonitorPod|RemoteMsgHeartbeat|
RemoteMsgGetSiteStatus|RemoteSiteRoleMonitoring|RemoteClusterPodFailure|RemoteMsgNotifyFailover|
RemoteMsgNotifyPrepareFailover',status=~'success|monitoring.*'}) by (namespace, AdminNode,

ControlActionType, ControlActionNameType,  pod, status)"
 labels     [ pod AdminNode ControlActionNameType status ]
 alias      gr-geo-monitoring
exit
bulk-stats query GR-Geo-Monitoring-Total
 expression "sum(geo_monitoring_total{ControlActionNameType=~'MonitorPod|RemoteMsgHeartbeat|
RemoteMsgGetSiteStatus|RemoteSiteRoleMonitoring|RemoteClusterPodFailure|RemoteMsgNotifyFailover
|RemoteMsgNotifyPrepareFailover|MonitorVip'})
by (namespace, AdminNode, ControlActionType, ControlActionNameType,  pod, status)"
 labels     [ pod AdminNode ControlActionNameType status ]
 alias      gr-geo-monitoring
exit
bulk-stats query GR-Geo-Replication-Failure
 expression
"sum(geo_replication_total{ReplicationNode=~'CACHE_POD|ETCD|PEER',status!='success',
ReplicationRequestType='Response'}) by (namespace, ReplicationNode, ReplicationSyncType,
ReplicationReceiver,ReplicationRequestType,status,status_code)"
 labels     [ pod ReplicationNode ReplicationReceiver ReplicationRequestType
ReplicationSyncType status status_code ]
 alias      gr-geo-replication-failure
exit
bulk-stats query GR-Geo-Replication-Success
 expression "sum(geo_replication_total{ReplicationNode=~'CACHE_POD|ETCD|PEER',
status='success',ReplicationRequestType='Response'}) by (namespace, ReplicationNode,
ReplicationSyncType,ReplicationReceiver,ReplicationRequestType,status)"
 labels     [ pod ReplicationNode ReplicationReceiver ReplicationRequestType
ReplicationSyncType status ]
 alias      gr-geo-replication-success
exit
bulk-stats query GR-Geo-Replication-Total
 expression "sum(geo_replication_total{ReplicationNode=~'CACHE_POD|ETCD|PEER'})
by (namespace, ReplicationNode, ReplicationSyncType,ReplicationReceiver,
ReplicationRequestType,pod)"
 labels     [ pod ReplicationNode ReplicationReceiver ReplicationRequestType
ReplicationSyncType ]
 alias      gr-geo-replication-total
exit
bulk-stats query GR-Trigger-ResetRole-Api
 expression "sum(geo_monitoring_total{ControlActionNameType=~'TriggerGRApi|ResetRoleApi'})

by (namespace, AdminNode, ControlActionType, ControlActionNameType,  pod, status,
status_code)"
 labels     [ pod AdminNode ControlActionNameType status status_code ]
 alias      gr-api
exit
bulk-stats query GR-CDL-Index-Replication
 expression "sum(consumer_kafka_records_total) by (pod, origin_instance_id)"
 labels     [ origin_instance_id pod ]
 alias      gr-cdl-index-replication
exit
bulk-stats query GR-CDL-Inter-Rack-Replications-Failures
 expression "sum(datastore_requests_total{local_request='0',errorCode!='0'}) by
```

```
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-inter-rack-replications
exit
bulk-stats query GR-CDL-Inter-Rack-Replications-Success
 expression "sum(datastore_requests_total{local_request='0',errorCode='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-inter-rack-replications
exit
bulk-stats query GR-CDL-Inter-Rack-Replications-Total
 expression "sum(datastore_requests_total{local_request='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-inter-rack-replications
exit
bulk-stats query GR-CDL-Intra-Rack-Operations-Failures
 expression "sum(datastore_requests_total{local_request='1',errorCode!='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-intra-rack-operations
exit
bulk-stats query GR-CDL-Intra-Rack-Operations-Success
 expression "sum(datastore_requests_total{local_request='1',errorCode='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-intra-rack-operations
exit
bulk-stats query GR-CDL-Intra-Rack-Operations-Total
 expression "sum(datastore_requests_total{local_request='1'}) by
(operation,sliceName,errorCode)"
 labels     [ errorCode operation sliceName ]
 alias      gr-cdl-intra-rack-operations
exit
bulk-stats query GR-CDL-Session-Count-Per-Slice
 expression
sum(avg(db_records_total{namespace=~'$namespace',session_type='total'})by(systemId,sliceName))by(sliceName)

 labels     [ sliceName ]
 alias      gr-cdl-session-count-per-slice
exit
bulk-stats query GR-CDL-Session-Count-Per-System-ID
 expression sum(avg(db_records_total{namespace=~'$namespace',session_type='total'})
by(systemId,sliceName))by(systemId)
 labels     [ systemId ]
 alias      gr-cdl-session-count-per-system-id
exit
bulk-stats query GR-CDL-Slot-Records-Per-Slice
 expression "sum(slot_records_total{pod=~'.*',systemId!=''}) by (pod, sliceName)"
 labels     [ pod sliceName ]
 alias      gr-cdl-slot-records-per-slice
exit
bulk-stats query GR-CDL-Slot-Records-Per-System-ID
 expression "sum(slot_records_total{pod=~'.*',systemId!=''}) by (pod, systemId)"
 labels     [ pod systemId ]
 alias      gr-cdl-slot-records-per-system-id
exit
bulk-stats query GR-CDL-Total-Session-Count
 expression "sum(db_records_total{namespace=~'$namespace',session_type='total'}) by
(systemId,sliceName)"
 labels     [ sliceName systemId ]
 alias      gr-cdl-total-session-count
exit
```

For more information on GR-related statistics, see the following:

- In cnSGW-C statistics, you can filter GR-specific statistics using gr_instance_id label.

  For more information, see the *UCC Serving Gateway Control Plane Function - Metrics Reference*.

Alerts

The following section provides details on GR alerts.

BFD Alerts

The following table list alerts for rule group BFD with *interval-seconds* as 60.

Table 213: Alert Rule Group - BFD

Alert Rule	Severity	Duration (in mins)	Type
BFD-Link-Fail	critical	1	Communication Alarm
	Expression: sum by (namespace,pod,status) (bgp_speaker_bfd_status{status='BFD_STATUS'}) == 0 **Description:** This alert is generated when BFD link associated with BGP peering is down.		

GR Alerts

The following table list alerts for rule group GR with *interval-seconds* as 60.

Table 214: Alert Rule Group - GR

Alert Rule	Severity	Duration (in mins)	Type
Cache-POD-Replication-Immediate-Local	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of CACHE_POD sync type:Immediate and replication receiver:Local is below threshold value.		

Alert Rule	Severity	Duration (in mins)	Type
Cache-POD-Replication-Immediate-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of CACHE_POD sync type:Immediate and replication receiver:Remote is below threshold value.		
Cache-POD-Replication-PULL-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='PULL',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='PULL',ReplicationReceiver='remote', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of CACHE_POD sync type:PULL and replication receiver:Remote is below threshold value.		
ETCD-Replication-Immediate-Local	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of ETCD sync type:Immediate and replication receiver:Local is below threshold value.		

Alert Rule	Severity	Duration (in mins)	Type
ETCD-Replication-Immediate-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of ETCD sync type:Immediate and replication receiver:Remote is below threshold value.		
ETCD-Replication-PULL-Remote	critical	1	Communication Alarm
	Expresion: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='PULL',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/ sum by (namespace) (increase(geo_replication_total {ReplicationNode='ETCD',ReplicationSyncType='PULL', ReplicationReceiver='remote',ReplicationRequestType= 'Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of ETCD sync type:PULL and replication receiver:Remote is below threshold value.		
Heartbeat-Remote-Site	critical	-	Communication Alarm
	Expression: sum by (namespace) (increase(geo_monitoring_total{ControlActionNameType= 'RemoteMsgHeartbeat',status!='success'}[1m])) > 0 **Description:** This alert is triggerd when periodic Heartbeat to remote site fails.		
Local-Site-POD-Monitoring	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='MonitorPod'}[1m])) > 0 **Description:** This alert is triggerd when local site pod monitoring failures breaches the configured threshold for the pod mentioned in Label:{{$labels.AdminNode}}.		

Alert Rule	Severity	Duration (in mins)	Type
PEER-Replication-Immediate-Local	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='PEER', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Response',status='success'} [1m]))/sum by (namespace) (increase(geo_replication_total {ReplicationNode='PEER',ReplicationSyncType= 'Immediate',ReplicationReceiver='local', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of PEER sync type:Immediate and replication receiver:Local is below threshold value.		
PEER-Replication-Immediate-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='PEER', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'} [1m]))/sum by (namespace) (increase(geo_replication_total {ReplicationNode='PEER',ReplicationSyncType='Immediate', ReplicationReceiver='remote',ReplicationRequestType= 'Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of PEER sync type:Immediate and replication receiver:Remote is below threshold value.		
RemoteCluster-PODFailure	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='RemoteClusterPodFailure'}[1m])) > 0 **Description:** This alert is generated when pod failure is detected on the Remote site for the pod mentioned in Label:{{$labels.AdminNode}}.		

Alert Rule	Severity	Duration (in mins)	Type
RemoteMsg NotifyFailover	critical	1	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='RemoteMsgNotifyFailover',status!='success'}[1m])) > 0 **Description:** This alert is generated when transient role RemoteMsgNotifyFailover has failed for the reason mentioned in Label:{{$labels.status}}.		
RemoteMsg NotifyPrepare Failover	critical	1	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='RemoteMsgNotifyPrepareFailover',status!='success'}[1m])) > 0 **Description:** This alert is generated when transient role RemoteMsgNotifyPrepareFailover has failed for the reason mentioned in Label:{{$labels.status}}.		
RemoteSite-RoleMonitoring	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='RemoteSiteRoleMonitoring'}[1m])) > 0 **Description:** This alert is generated when RemoteSiteRoleMonitoring detects role inconsistency for an instance on the partner rack and accordingly changes the role of the respective instance on local rack to Primary. The impacted instance is in Label:{{$labels.AdminNode}}.		
ResetRoleApi -Initiated	critical	-	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='ResetRoleApi'}[1m])) > 0 **Description:** This alert is generated when ResetRoleApi is initiated with the state transition of roles mentioned in Label:{{$labels.status}}.		
TriggerGRApi -Initiated	critical	-	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='TriggerGRApi'}[1m])) > 0 **Description:** This alert is generated when TriggerGRApi is initiated for the reason mentioned in Label:{{$labels.status}}.		

Alert Rule	Severity	Duration (in mins)	Type
VIP-Monitoring -Failures	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='MonitorVip'}[1m])) > 0 **Description:** This alert is generated when GR is generated upon detecting VIP monitoring failures for the VIP and Instance mentioned in the Label:{{$labels.AdminNode}}.		

CDL Alerts

The following table list alerts for rule group CDL with *interval-seconds* as 60.

Table 215: Alert Rule Group - CDL

Alert Rule	Severity	Duration (in mins)	Type
GRPC-Connections-Remote-Site	critical	1	Communication Alarm
	Expression: sum by (namespace, pod, systemId) (remote_site_connection_status) !=4 **Description:** This alert is generated when GRPC connections to remote site are not equal to 4.		
Inter-Rack -CDL-Replication	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(datastore_requests_total{local_request=\"0\", errorCode=\"0\"}[1m]))/sum by (namespace) (increase(datastore_requests_total{local_request=\"0\"} [1m])))*100 < 90 **Description:** This alert is generated when the Inter-rack CDL replication success rate is below threshold value.		
Intra-Rack -CDL-Replication	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(datastore_requests_total{local_request=\"1\", errorCode=\"0\"}[1m]))/sum by (namespace) (increase(datastore_requests_total{local_request=\"1\"} [1m])))*100 < 90 **Description:** This alert is generated when the Intra-rack CDL replication success rate is below threshold.		

Maintenance Mode

cnSGW-C supports the maintenance mode flag to disable the impact on a cluster if the cluster in GR setup is scheduled for in-service (rolling upgrade). This is useful so that the other mated cluster executes its responsibility and other activities on the targeted cluster without any issue.

If the maintenance mode flag is set to **true**, cluster role change and GR trigger for the rack is allowed only in case of CLI-based failover.

During the execution, all the monitoring threads check the runtime value for the flag and hold the execution if the maintenance mode flag is set to **true**. By default, for fresh installation, the flag is set to **false**. Based on your requirements, to configure the maintenance mode, use the following configuration.

```
config
  geo maintenance mode { true | false }
  end
```

NOTES:

- **geo maintenance mode { true | false }** - Enable/disable the maintenance mode.

  The value for the maintenance mode is stored in.`etcd`

Both the clusters can be under maintenance at the same time. You can push the system in maintenance mode if the mated cluster is already under maintenance. Before you start the maintenance activity, set the `geo maintenance mode` flag value to **true**. When maintenance is complete, reset the flag to **false** after confirming the health of the system.

When the maintenance flag is set to true:

- All the monitoring activities are paused.
- The mated cluster cannot trigger the failover when it detects the local failure.
- Replication activities continue on the cluster.
- Maintenance mode doesn't change instance roles of the site implicitly. However, role change is possible using `geo switch-role role` CLI command.

GR trigger is not allowed towards and from the cluster under maintenance. Only CLI-based failover is supported from the cluster under the maintenance. After disabling maintenance mode, start with new data for pod and VIP monitoring. Remote cluster is informed about the maintenance mode value using the `NotifyMaintenanceActivity() [Operation 24]` message.

Example

The following is a configuration example:

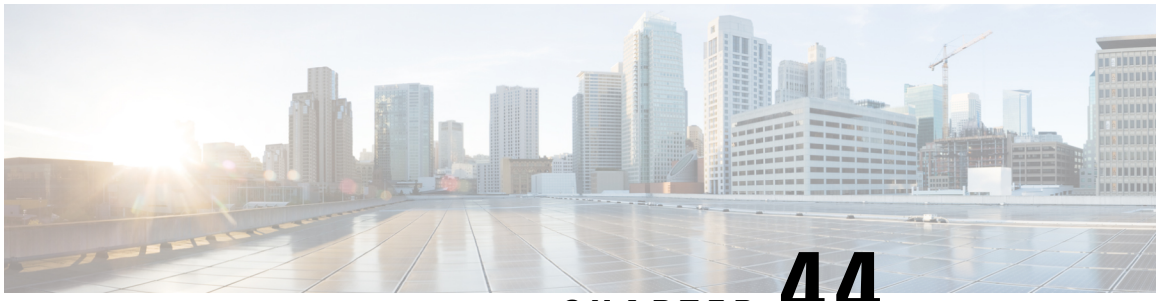
```
geo maintenance mode true
result "success"

geo maintenance mode false
result "success"
```

Viewing the Maintenance Mode Status

To check the maintenance mode status, use the following `show` command.

```
show geo maintenance mode
result "geo maintenance mode is disabled"
```



CHAPTER 44

Service Configuration Enhancements

- Feature Summary and Revision History, on page 603
- Feature Description, on page 603
- Feature Configuration, on page 604
- Troubleshooting Information, on page 609

Feature Summary and Revision History

Summary Data

Table 216: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 217: Revision History

Revision Details	Release
Support added for DNN List	2021.02.3
First introduced	2020.01.0

Feature Description

cnSGW-C supports Subscriber Map and Operator Policy configurations.

SGW profile represents SGW-service or node. The operator policy is decided based on subscriber policy association.

Feature Configuration

Configuring this feature involves the following steps:

- SGW profile. For more information, see Configuring the SGW Profile, on page 604.
- Subscriber policy. For more information, see Configuring the Subscriber Policy, on page 605.
- Operator policy. For more information, see Configuring the Operator Policy, on page 606.
- Policy DNN. For more information, see Configuring the Policy DNN, on page 606.

Configuring the SGW Profile

To configure this feature, use the following configuration:

```
config
   profile sgw sgw_name
      locality locality_code
      fqdn dnn_name
      subscriber-policy policy_name
      end
```

NOTES:

- **locality** *locality_code*—Specify the locality code. Must be a string.
- **fqdn** *dnn_name*—Specify the cnSGW-C FQDN.
- **subscriber-policy** *policy_name*—Specify the subscriber policy name. Must be a string.

Configuration Example

The following is an example configuration.

```
config
   profile sgw sgw-data
      locality LOC1
      fqdn 209.165.200.254
      subscriber-policy subpol1
      end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw
profile sgw sgw-data
locality LOC1
fqdn 209.165.200.254
subscriber-policy subpol1
```

Configuring the Subscriber Policy

Note The maximum number of supported subscriber map profiles is 64.

To configure this feature, use the following configuration:

```
config
   policy subscriber subscriber_name
      precedence precedence_value
         imsi
            mcc mcc_value
            mnc mnc_value
            msin first_value last_value
         serving-plmn
            mcc mcc_value
            mnc mnc_value
         imsi-start-range range_value
         imsi-stop-range range_value
         supi-start-range range_value
         supi-stop-range range_value
         operator-policy policy_name
         end
```

Note All parameters are optional.

NOTES:

- **precedence** *precedence_value*—Specify the precedence for entry. Must be an integer in the range of 1-2048.
- **mcc** *mcc_value*—Specify the Mobile Country Code (MCC). Must be a three digit integer.
- **mnc** *mnc_value*—Specify the Mobile Network code (MNC). Must be a two or three digit integer.
- **msin** *first_value last_value*—Specify the mobile subscriber identification number (MSIN) range.

  first_value—Specify starting value of the MSIN range. Must be an integer in the range of 1-9999999999.

  last_value—Specify the ending value of the MSIN range. Must be an integer in the range of 1-9999999999.
- **operator-policy** *policy_name*—Specify the operator policy name. Must be a string.
- **imsi-start-range** *range_value*—Specify the IMSI start range. Must be an integer in the range of 100000000000000-9999999999999999.
- **imsi-stop-range** *range_value*—Specify the IMSI stop range. Must be an integer in the range of 100000000000000-9999999999999999.
- **supi-start-range** *range_value*—Specify the SUPI start range. Must be an integer in the range of 100000000000000-999999999999999.

- **supi-stop-range** *range_value*—Specify the SUPI stop range. Must be an integer in the range of 100000000000000-999999999999999.

Configuration Example

The following is an example configuration.

```
config
   policy subscriber sub1
      precedence 2
         imsi mcc 123 mnc 456
         imsi msin first 99 last 100
      serving-plmn mcc 404 mnc 678
      supi-start-range 100000000000001
      supi-stop-range  199999999999999
      imsi-start-range 200000000000001
      imsi-stop-range  299999999999999
      operator-policy op1
      end
```

Configuring the Operator Policy

Note The maximum number of supported operator policy profiles is 1000.

To configure this feature, use the following configuration:

```
config
  policy operator operator_name
  policy dnn dnn_policy_name
  end
```

NOTES:

- **policy operator** *operator_name*—Specify the operator policy name. Must be a string.
- **policy dnn** *dnn_policy_name*—Specify the DNN policy name. Must be a string.

Configuration Example

The following is an example configuration.

```
config
   policy operator op1
   policy dnn pol1
   end
```

Configuring the Policy DNN

This section describes how to configure Policy DNN and adding it to cnSGW-C. The DNN support enables you to determine the exact APN profile as per the APN name, APN network-identifier and APN operator-identifier.

Note The maximum number of supported DNN policies is 1000.

To configure this feature, use the following configuration:

```
config
   policy dnn dnn_policy_name
       dnn dnn_name
          dnn-list dnn_list
          profile profile_name
       dnn network-identifier network_identifier_name operator-identifier
operator_identifier_name profile profile_name
       dnn operator-identifier operator_identifier_name profile profile_name
       dnn operator-identifier profile profile_name
       end
```

NOTES:

- **dnn** *dnn_name*—Specify the DNN name.
- **network-identifier** *network_identifier_name*—Specify the network identifier. Must be a string.
- **profile** *profile_name*—Specify the profile name. Must be a string.
- **operator-identifier** *operator_identifier_name*—Specify the operator identifier. Must be a string.
- **profile** *default_dnn_profile*—Specify the default DNN profile name.
- **dnn-list** *dnn_list*—Specify the DNN list for selecting a DNN profile. It helps in minimizing the entries needed for pointing multiple DNN to a single profile.

Note With present evaluation criteria, following is the matching order to select the associated profile:

- DNN/DNN List
- NI+OI
- NI
- OI
- Default

Don’t configure overlapping criteria.

Configuration Example

The following is an example configuration.

```
config
   policy dnn polsub1
      dnn network-identifier ims profile ims1
      dnn network-identifier ims operator-identifier ims.com profile ims
      dnn network-identifier voice operator-identifier volte profile voiceprofile
```

```
      dnn operator-identifier data profile data-profile
   profile default-dnn-profile
   end
```

```
config
   policy dnn polsub1
      dnn intershat dnn-list [ intershat1 intershat2 intershat3 ]
         profile profile_name
            dnn network-identifier ims operator-identifier ims.com profile ims
            dnn operator-identifier volte profile voiceprofile
            dnn operator-identifier data profile profile_name
            end
```

Configuration Modification Impact

This section describes the impact or behavior of configuration change on existing call, new PDN, or new subscriber.

Modification	cnSGW-C Existing Call	cnSGW-C New PDN or New subscriber
Define a new SGW-profile and delete the old profile (with or without the pod restart)	When the new transaction happens, the call gets loaded on cn-SGW-C from CDL. *Observation:* Change in the cnSGW-C profile configuration and termination of the call.	Rejects the new PDN and deletes the existing call. New subscriber uses new SGW profile.
Delete the subscriber map	No impact	Applies modified configuration for the following: • subscriber policy • operator policy • dnn policy • dnn profile
Modify the subscriber map in SGW-service	No impact	Applies modified configuration for the following: • subscriber policy • operator policy • dnn policy • dnn profile

Modification	cnSGW-C Existing Call	cnSGW-C New PDN or New subscriber
Delete the operator policy	No impact	Applies modified configuration for the following: • subscriber policy • operator policy • dnn policy • dnn profile
Add the deleted or new operator policy	No impact	Applies modified configuration for the following: • subscriber policy • operator policy • dnn policy • dnn profile
Modify the operator policy name in the subscriber map	No impact	Applies modified configuration for the following: • subscriber policy • operator policy • dnn policy • dnn profile

Troubleshooting Information

This section describes the troubleshooting information that enables you to resolve the service configuration issues.

Configuration Errors

This section describes the errors that cnSGW-C reports during the service configuration.

Subscriber Policy Configuration Errors

```
show config-error
ERROR
COMPONENT ERROR DESCRIPTION
--------------------------------------------------------------------------
SGWProfile Subscriber policy name : polSubSgw in profile sgw1 is not configured
```

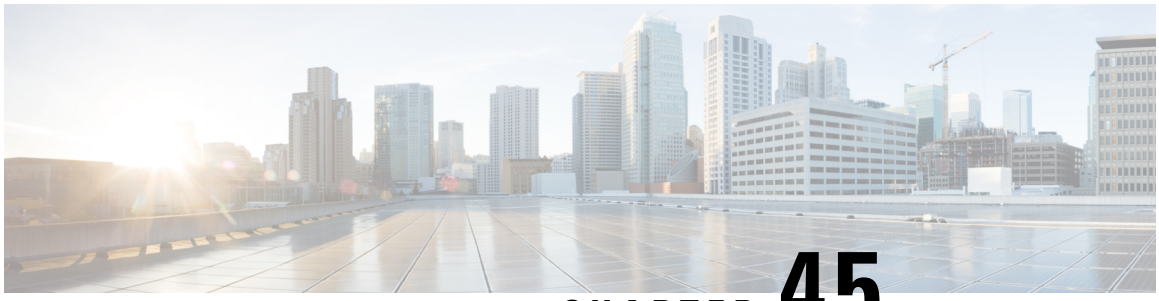
Operator Policy Configuration Errors

```
show config-error
ERROR COMPONENT ERROR DESCRIPTION
```

```
------------------------------------------------------------------------
SubscriberPolicy Operator policy : op2 under subscriber policy polSubSgw is not configured
```

DNN Policy Configuration Errors

```
show config-error
ERROR COMPONENT ERROR DESCRIPTION
------------------------------------------------------------------------
OperatorPolicy Dnn policy name : dnn_1 in operator policy op1 is not configured
```



CHAPTER 45

SGW Charging Support

- Feature Summary and Revision History, on page 611
- Feature Description, on page 612
- How it Works, on page 614
- Feature Configuration, on page 630
- CDR Fields Supported in cnSGW-CDRs, on page 641
- SGW Charging OAM Support, on page 656

Feature Summary and Revision History

Summary Data

Table 218: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 219: Revision History

Revision Details	Release
First introduced.	2021.02.0

Feature Description

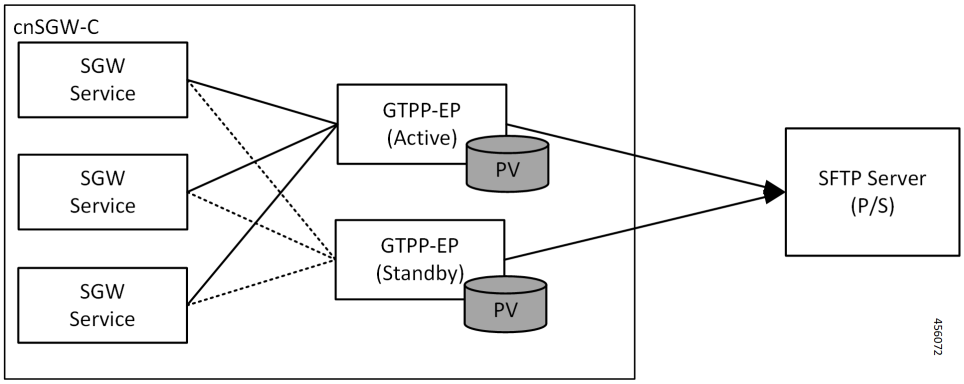
Table 220: Feature History

Feature Name	Release Information	Description
Charging Support for Converged Calls	2024.02.0	With UPF supporting the collapsed-data path functionality, cnSGW supports charging with converged UPF (UPF + SGW-U). This support prevents affecting the Local Breakout (LBO) calls for which the carrier uses SGW-based CDRs for reporting and charging. **Default Setting**: Not Applicable
Custom3 file format for SGW CDR	2025.04.0	This feature adds support for the custom3 file format for SGW Charging Data Records (CDRs), giving operators more flexibility in how they store and manage CDRs. In addition to the existing custom1 and custom5 formats, operators can configure the system to use custom3 through a CLI setting under the gtpp-profile. To improve operational reliability, the system can automatically transfer these custom3 CDR files to external storage servers using encrypted SFTP, with built-in failover support to handle any connection issues. New CLI commands make it easier for operators to view, manage, and troubleshoot CDR files in the custom3 format.

cnSGW-c supports the following:

- GTPP charging (Gz) interface
- Monitor subscriber for Charging Data Record (CDR)
- CDR dictionary: **custom24**
- Three custom file formats: **custom1** (default) , **custom5** and **custom3**
- One replica of GTPP-EP pod which is functional with active or standby mode (two pods get spawned when GTPP-EP pod configured with one instance)
- Writing CDR records to the local file system
- Charging with converged UPF (UPF + SGW-U)

Architecture



- GTP’ (GTP Prime) or GTPP-EP is the new endpoint pod and interfaces with mediation or CGF server over SFTP
- GTP’ attaches to the local disk (Persistent Volume). This attachment is with each server or virtual machine (VM)
- SGW-service generates CDRs and sends the records to the GTP' endpoint for the storage
- GTP' stores the CDRs in ASN.1 encoding in flat files in persistent storage
- GTP' pushes the flat files over SCTP towards the mediation server or CGF

The charging functionality is split into two parts.

- Accounting and CDR generation:
  - SGW-service generates usage reporting rule (URR) for each established bearer on the Sxa interface with SGW-U
  - SGW-service uses the reported usage information with the trigger event to generate accounting information
- CDR management and storage:
  - GTPP-EP microservice or K8 pod archives the CDRs and pushes the CDR files to the external storage server
  - GTPP-EP receives the proto-CDRs from SGW-service over the streaming GRPC IPC endpoint
  - GTPP-EP encodes each received proto-CDR into ASN.1 format as specified in the dictionary (from CLI)
  - The ASN.1 CDRs are written to flat files in the specified pattern as specified in the CLI configuration
  - Transfers to the new CDR files to the configured external storage server using SFTP protocol periodically

Roaming Support

Roaming scenarios uses a Gz interface and offline accounting functions to match the CDR records with the foreign PGW.

The operator policy provides mechanisms to modify the behavior of subsets of subscribers described in the SGW profile. cnSGW-C supports call-control-profile under the operator-policy to control the accounting mode (enable or disable the charging) and define more charging configurations.

The default accounting mode is NONE which indicates charging is disabled.

The accounting mode value from the call control profile overrides the configured value in the SGW profile.

See the following configuration details:

- Call Control Profile Configuration
- Charging-Characteristics under Call-Control-Profile

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

URR Installation on Initial Attach Call Flow

This section describes URR Installation on Initial Attach call flow.

Figure 113: URR Installation on Initial Attach Call Flow

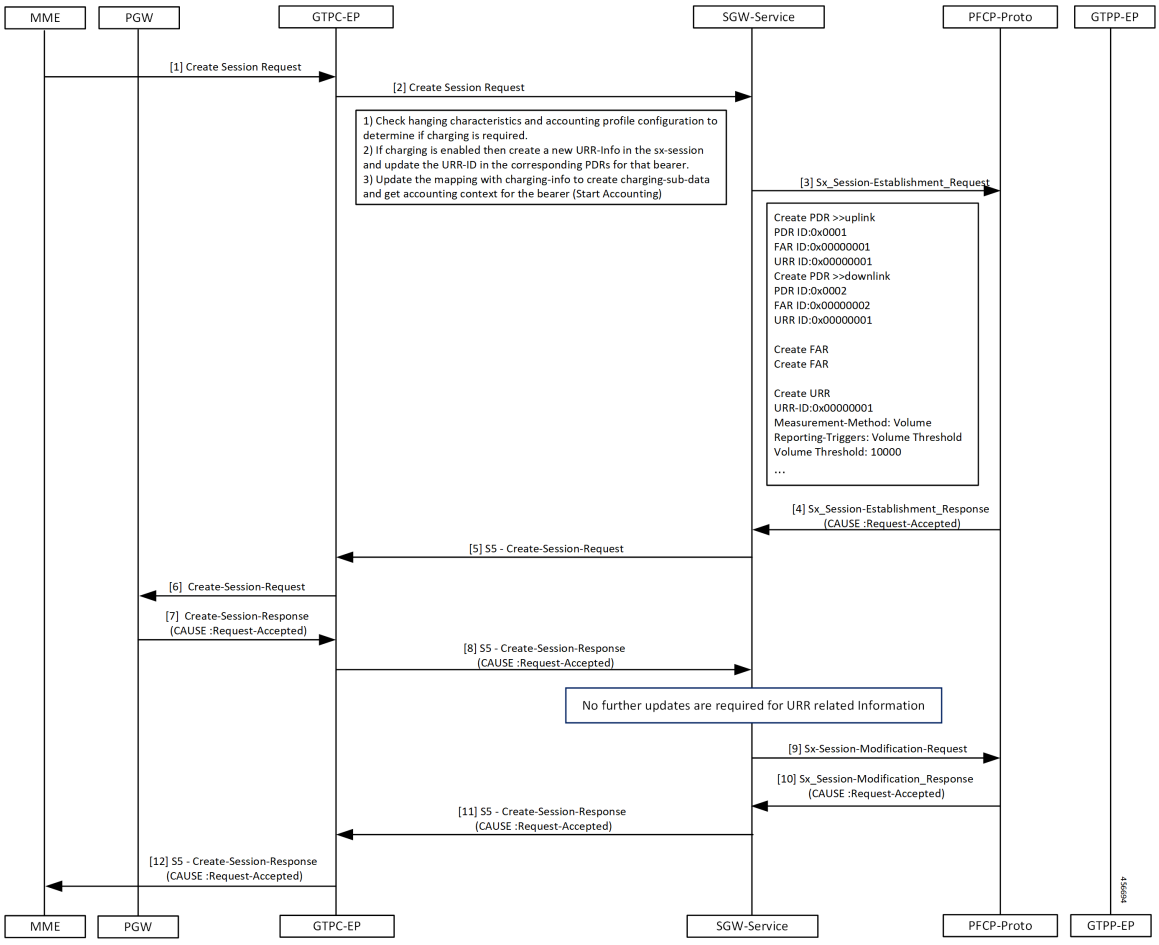


Table 221: URR Installation on Initial Attach Call Flow Description

Step	Description
1	The MME sends the Create Session Request to the GTPC-EP.
2	The GTPC-EP forwards the Create Session Request to the SGW-service pod.
3	The SGW-service pod sends the Sx Session Establishment Request to the PFCP proto
4	The PFCP proto sends the Sx Session Establishment Response to the SGW-service with the cause as Request-Accepted.
5	The SGW-service pod sends the S5 Create Session Request to the GTPC-EP.
6	The GTPC-EP sends the S5 Create Session Request to the PGW.
7	The PGW sends the Create Session Response to the GTPC-EP with the cause as Request-Accepted.
8	The GTPC-EP sends the S5 Create Session Response to the SGW-service with the cause as Request-Accepted.

Step	Description
9	The SGW-service pod sends the Sx Session Modification Request to the PFCP proto.
10	The SGW-service pod receives the Sx Session Modification Response from the PFCP proto with the cause as Request-Accepted.
11	The SGW-service pod sends the S5 Create Session Response to the GTPC-EP with the cause as Request-Accepted.
12	The GTPC-EP forwards the S5 Create Session Response to the MME with the cause as Request-Accepted.

SGW CDR Call Flow

This section describes the SGW CDR call flow.

Figure 114: SGW CDR Call Flow

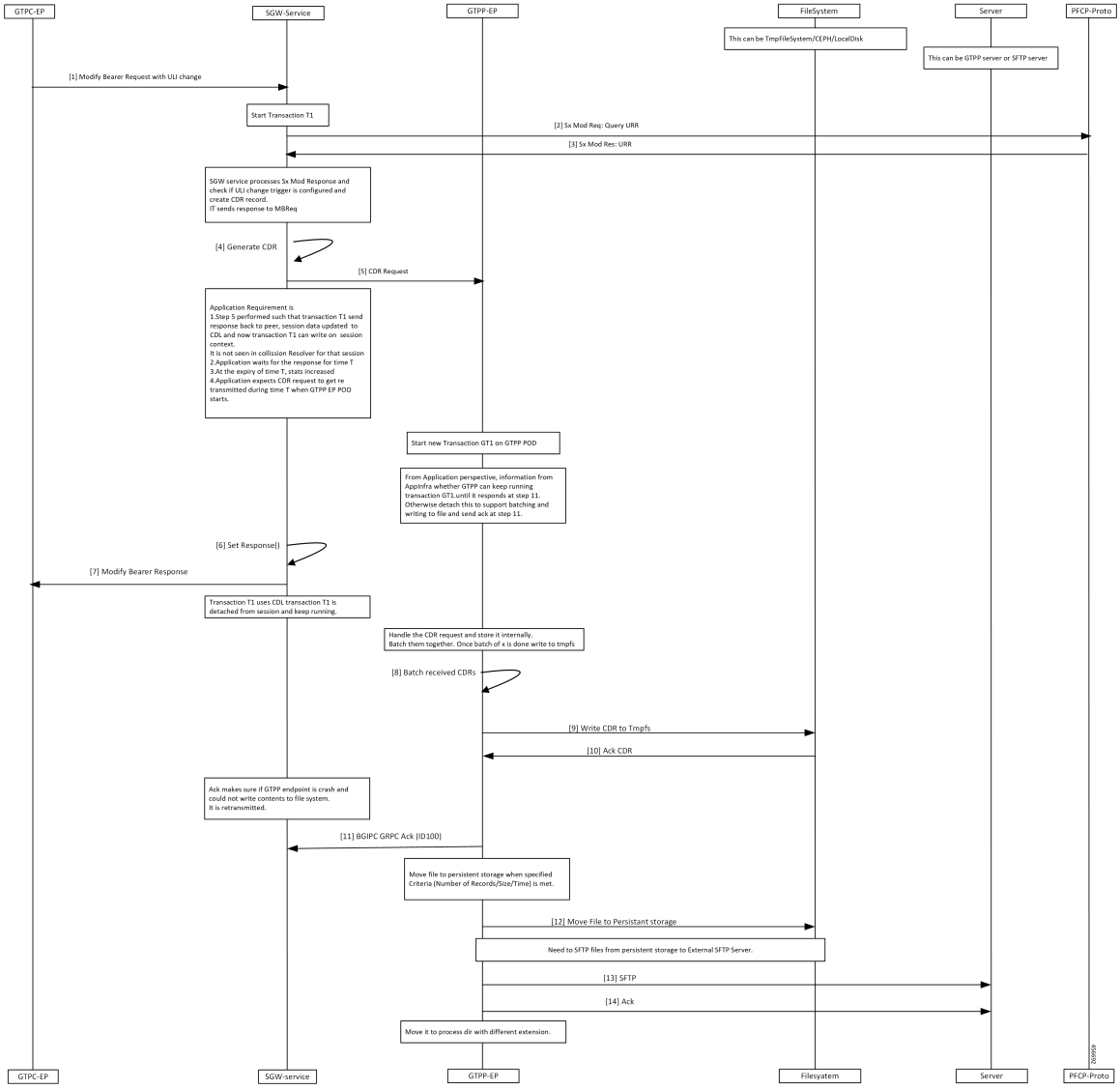


Table 222: SGW CDR Call Flow Description

Step	Description
1	The GTPC-EP sends the Modify Bearer Request with ULI to the SGW-service pod.
2	The SGW-service pod sends the Sx Mod Request with Query URR to the PFCP proto.
3	The SGW-service pod receives the Sx Mod Response with URR from the PFCP proto.
4	The SGW-service pod generate CDR.
5	The SGW-service pod sends the CDR request to the GTPP-EP.
6	The SGW-service pod triggers a set response () function.

Step	Description
7	The SGW-service pod sends the Modify Bearer Response to the GTPC-EP.
8	The GTPP-EP batches the received CDR requests .
9	The GTPP-EP sends the batched CDR requests to the TmpF5 file.
10	The GTPP-EP receives the CDR ACK from the file system.
11	The GTPP-EP sends GRPC ACK to the SGW-service.
12	The GTPP-EP moves the file to persistent storage when specified criteria (number of records or size or time) meets.
13	The GTPP-EP sends SFTP files from persistent storage to the server.
14	The GTPP-EP receives ACK from the server and moves it to the process directory with different extension.

URR Removal and CDR Reporting on Detach Call Flow

This section describes URR Removal and CDR Reporting on Detach call flow.

Figure 115: URR Removal and CDR Reporting on Detach Call Flow

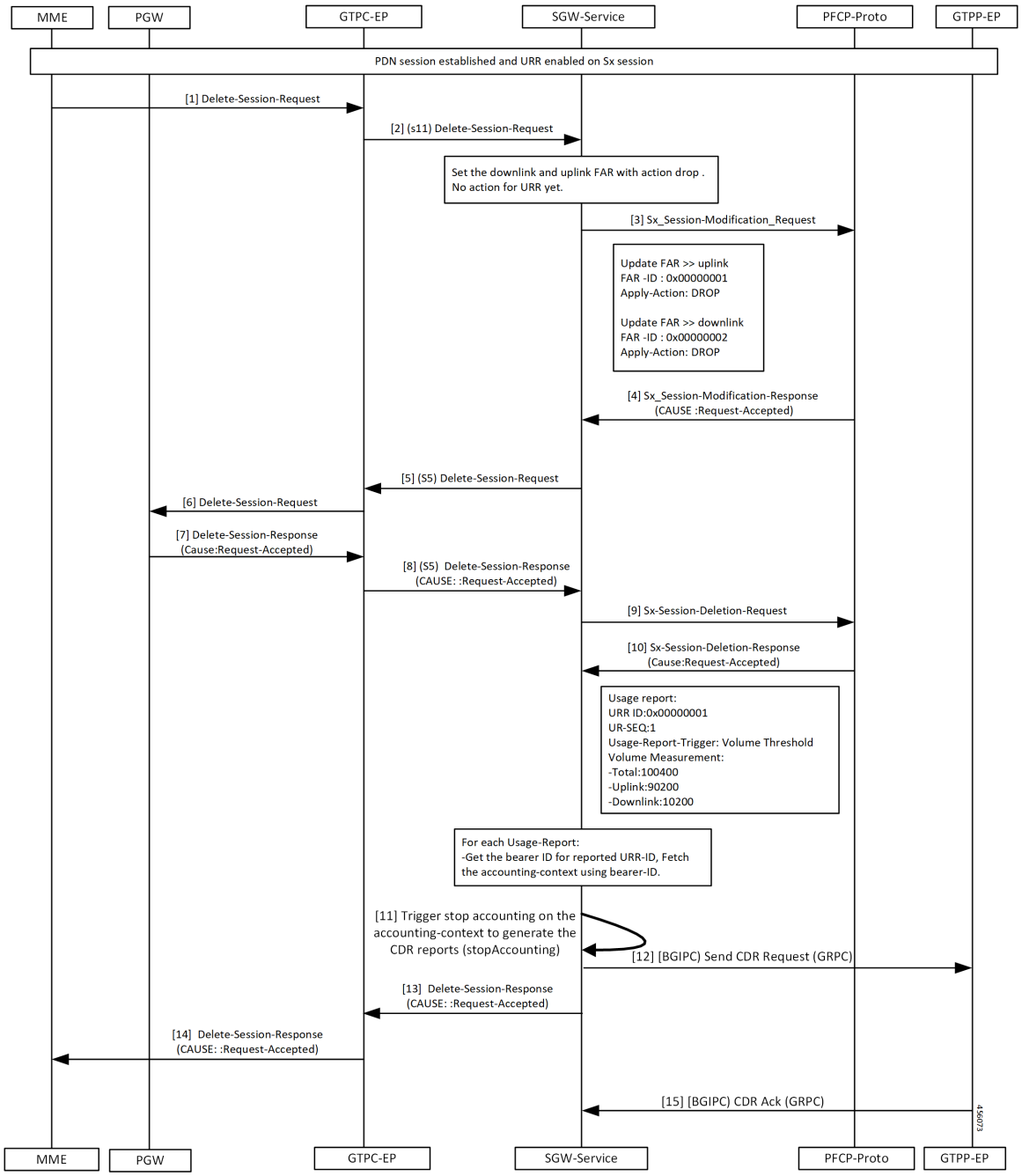


Table 223: URR Removal and CDR Reporting on Detach Call Flow Description

Step	Description
1	A PDN session is established and URR is enabled for Sx session. The MME sends the Delete Session Request to the GTPC-EP.

Step	Description
2	The GTPC-EP forwards the S11 Delete Bearer Request to the SGW-service pod.
3	The SGW-service pod sends the Sx Session Modification Request to the PFCP proto.
4	The PFCP proto sends the Sx Session Modification Response to the SGW-service pod with the cause as Request-Accepted.
5	The SGW-service pod sends the S5 Delete Session Request to the GTPC-EP.
6	The GTPC-EP sends the Delete Session Request to the PGW.
7	The PGW sends the Delete Session Response to the GTPC-EP with the cause as Request-Accepted.
8	The GTPC-EP sends the S5 Delete Session Response to the SGW-service pod with the cause as Request-Accepted.
9	The SGW-service pod sends the Sx Session Delete Request to the PFCP proto.
10	The SGW-service pod receives the Sx Session Delete Response from the PFCP proto with the cause as Request-Accepted.
11	The SGW-service pod triggers the CDR generation.
12	The SGW-service pod sends the CDR request to the GTPP-EP.
13	The SGW-service pod sends the Delete Session Response to the GTPC-EP with the cause as Request-Accepted.
14	The GTPC-EP pod forwards the Delete Session Response to the MME with the cause as Request-Accepted.
15	The GTPP-EP sends the CDR ACK to the SGW-service pod.

Usage Report on Hitting Threshold Call Flow

This section describes Usage Report on Hitting Threshold call flow.

Figure 116: Usage Report on Hitting Threshold Call Flow

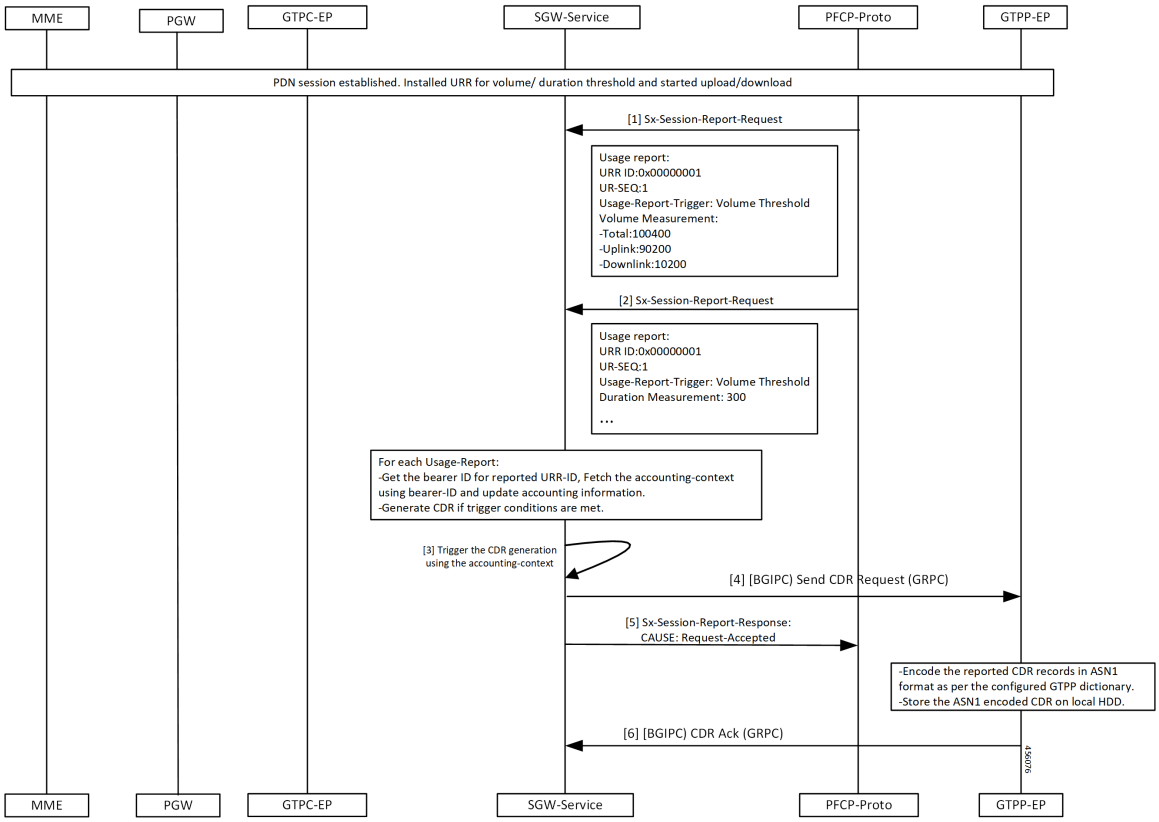


Table 224: Usage Report on Hitting Threshold Call Flow Description

Step	Description
1	Established a PDN session. Installed URR for the threshold duration. Trigger upload and download. The PFCP proto sends the Sx Session Report Request to the SGW-service pod.
2	The PFCP proto sends the Sx Session Report Request to the SGW-service pod until it reaches the threshold value of the User-plane.
3	The SGW-service pod triggers the CDR generation.
4	The SGW-service pod sends the CDR request to the GTPP-EP.
5	The SGW-service pod sends the Sx Session Report Response to the PFCP proto with the cause as Request-Accepted.
6	The GTPP-EP sends the CDR ACK to the SGW-service pod.

URR Installation for Dedicated Bearer Call Flow

This section describes the URR Installation for Dedicated Bearer call flow.

Figure 117: URR Installation for Dedicated Bearer Call Flow

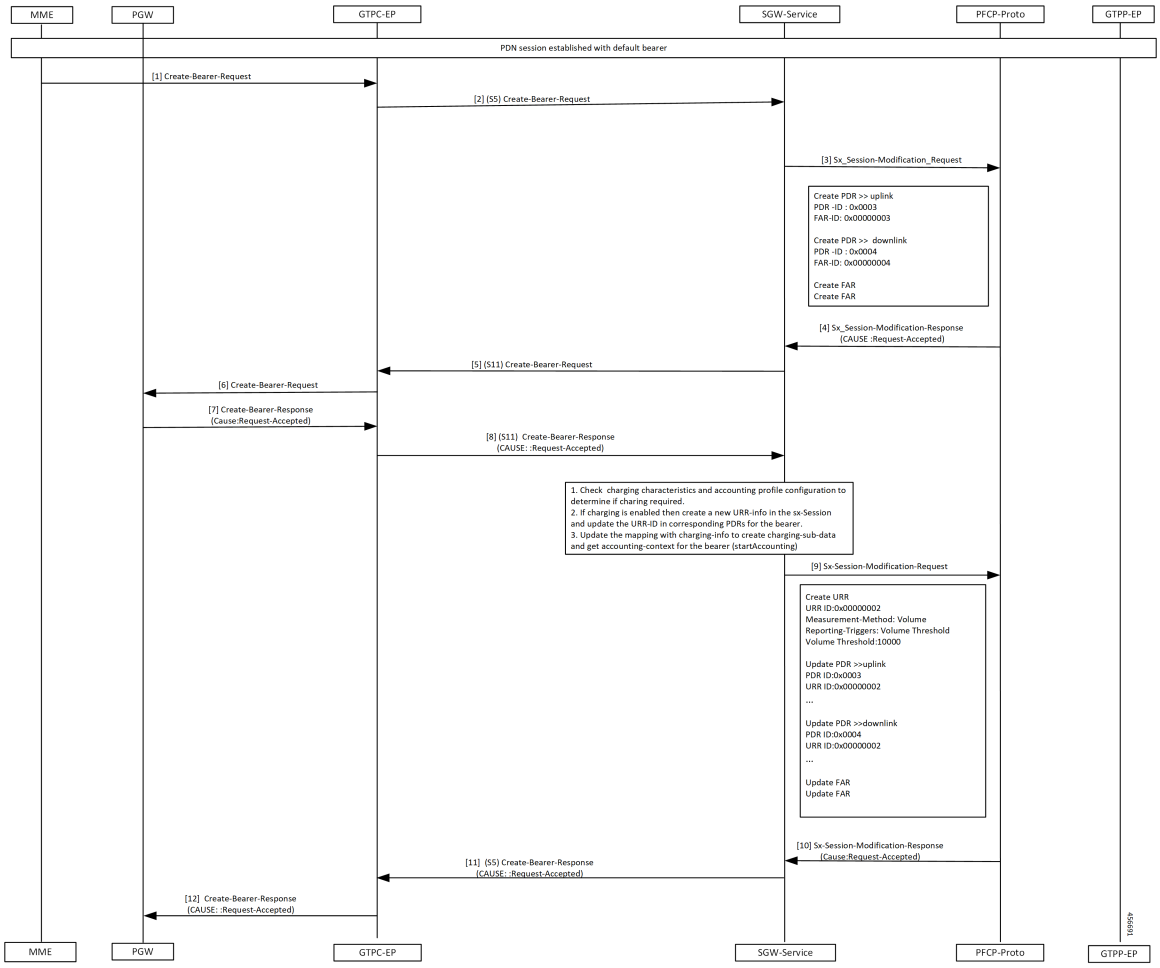


Table 225: URR Installation for Dedicated Bearer Call Flow Description

Step	Description
1	Established a PDN session with a default bearer. The PGW sends the Create Bearer Request to the GTPC-EP.
2	The GTPC-EP forwards the S5 Create Bearer Request to the SGW-service pod.
3	The SGW-service pod sends the Sx Session Modification Request to the PFCP proto.
4	The PFCP proto sends the Sx Session Modification Response to the SGW-service pod with the cause as Request-Accepted.
5	The SGW-service pod sends the S11 Create Bearer Request to the GTPC-EP.

Step	Description
6	The GTPC-EP forwards the S11 Create Bearer Request to the MME.
7	The GTPC-EP receives the Create Bearer Response to the GTPC-EP with the cause as Request-Accepted.
8	The GTPC-EP forwards the S11 Create Bearer Response to the SGW service with the cause as Request-Accepted.
9	The SGW-service pod sends the Sx Session Modification Request to the PFCP proto.
10,	The PFCP proto sends the Sx Session Modification Response to the SGW-service pod with the cause as Request-Accepted.
11	The SGW-service pod sends the S5 Create Bearer Response to the GTPC-EP with the cause as Request-Accepted.
12	The GTPC-EP sends the Create Bearer Response to the PGW with the cause as Request-Accepted.

URR Removal and CDR Generation on Deletion of Dedicated Bearer Call Flow

This section describes the URR Removal and CDR Generation on Deletion of Dedicated Bearer call flow.

Figure 118: URR Removal and CDR Generation on Deletion of Dedicated Bearer Call Flow

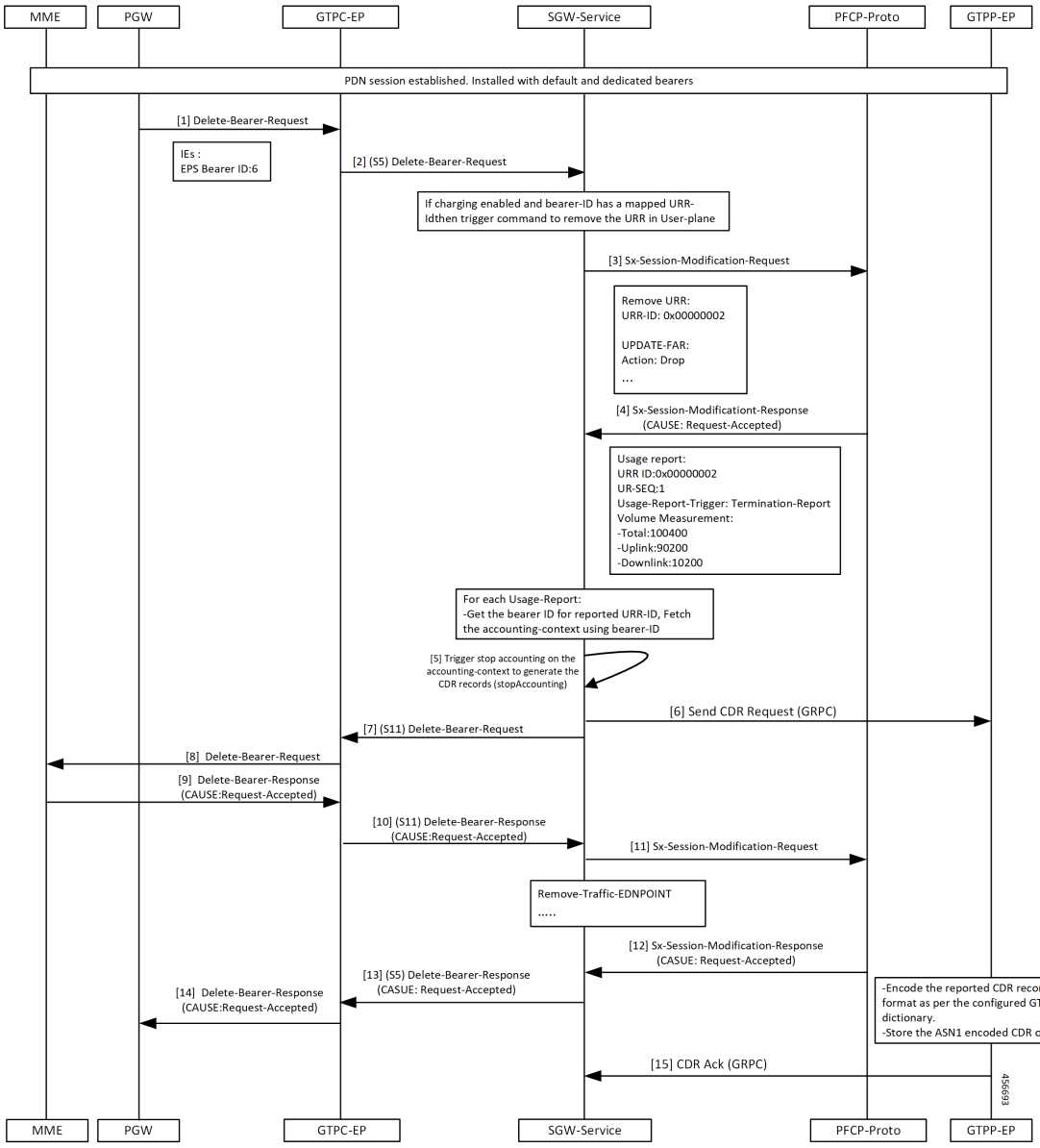


Table 226: URR Removal and CDR Generation on Deletion of Dedicated Bearer Call Flow Description

Step	Description
1	Established a PDN session with the default and dedicated bearer. The PGW sends the Delete Bearer Request to the GTPC-EP.
2	The GTPC-EP forwards the S5 Delete Bearer Request to the SGW-service pod.

Step	Description
3	The SGW-service pod requests a usage report query when charging enabled with QoS trigger and QoS change detected. The SGW-service pod sends the Sx Session Modification Request to the PFCP-Proto.
4	The PFCP proto sends the Sx Session Modification Response to the SGW-service pod with the cause as Request-Accepted.
5	The SGW-service pod triggers the CDR generation and sends CDR request to the GTPP-EP.
6	The SGW-service pod sends the S5 Sx Modify Bearer Request to the GTPP-EP.
7	The SGW-service pod sends the S11 Delete Bearer Request to the GTPC-EP.
8	The GTPC-EP forwards the Delete Bearer Request to the MME.
9	The GTPC-EP receives the Delete Barer Response from the MME with the cause as Request-Accepted.
10	The GTPC-EP forwards the S11 Delete Bearer Response to the SGW-service pod with the cause as Request-Accepted.
11	The SGW-service pod sends the Sx Session Modification Request to the PFCP proto.
12	The SGW-service pod receives the Sx Session Modification Response from the PFCP proto with the cause as Request-Accepted.
13	The SGW-service pod sends the S5 Delete Bearer Response to the GTPC-EP with the cause as Request-Accepted.
14	The GTPC-EP sends the Delete Bearer Response to the PGW with the cause as Request-Accepted.
15	The PFCP proto sends the CDR ACK to the SGW-service pod.

Volume Reporting on S11 Trigger Call Flow

This section describes Volume Reporting on S11 Trigger call flow.

Figure 119: Volume Reporting on S11 Trigger Call Flow

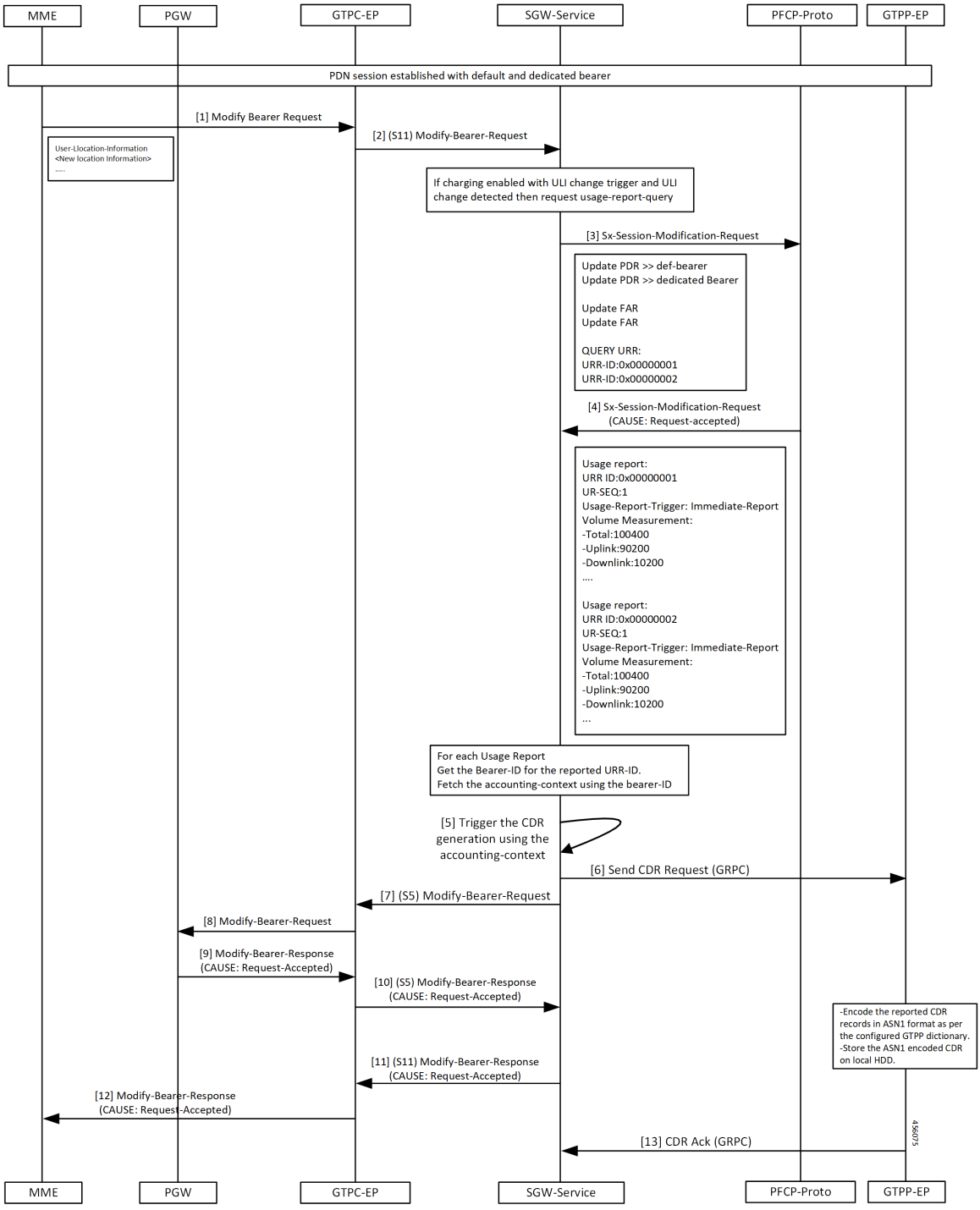


Table 227: Volume Reporting on S11 Trigger Call Flow Description

Step	Description
1	Established a PDN session with the default and dedicated bearers. The MME sends Modify Bearer Request to the GTPC-EP.
2	The GTPC-EP forwards the S11 Modify Bearer Request to the SGW-service pod.
3	The SGW-service pod requests the usage report query when charging enabled with QoS trigger and QoS change detected. The SGW-service pod sends the Sx session Modification Request to the PFCP proto.
4	The PFCP proto sends the Sx Session Modification Response to the SGW-service pod with the cause as Request-Accepted.
5	The SGW-service pod triggers the CDR generation.
6	The SGW-service pod sends the generated CDR request to the GTPP-EP.
7	The SGW-service pod sends the S5 Sx Modify Bearer Request to the GTPC-EP.
8	The GTPC-EP sends the Modify Bearer Request to the PGW.
9	The GTPC-EP receives the Modify Bearer Response from the PGW with the cause as Request-Accepted.
10	The GTPC-EP forwards the S5 Modify Bearer Response to the SGW-service pod with the cause as Request-Accepted.
11	The SGW-service pod sends the S11 Modify Bearer Response to the PGW with the cause as Request-Accepted.
12	The GTPC-EP sends the Modify Bearer Response to the MME with the cause as Request-Accepted.
13	The GTPP-EP sends the CDR ACK to the SGW-service pod.

Volume Reporting on S5 Trigger Call Flow

This section describes the Volume Reporting on S5 Trigger call flow.

Figure 120: Volume Reporting on S5 Trigger Call Flow

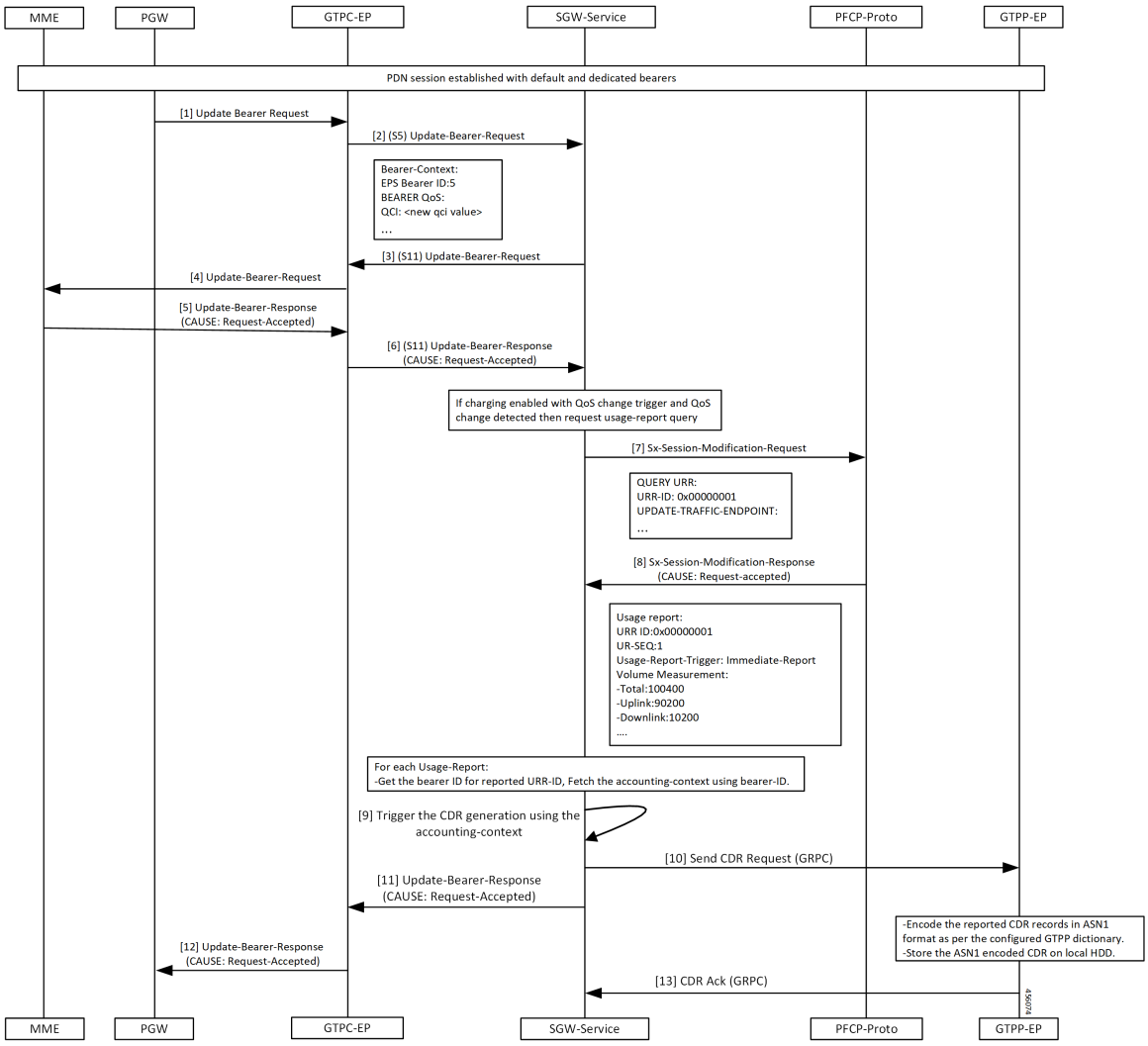


Table 228: Volume Reporting on S5 Trigger Call Flow Description

Step	Description
1	Established a PDN session with the default and dedicated bearers. The PGW sends the Update Bearer Request to the GTPC-EP.
2	The GTPC-EP forwards S5 Update Bearer Request to the SGW-service pod.
3	The SGW sends the Update Bearer request S11 to the GTPC-EP.
4	The GTPC-EP forwards the Update Bearer request to the MME.
5	The MME sends the Update Bearer Response to the GTPC-EP with the cause as Request-Accepted.

Step	Description
6	The GTPC-EP sends the S11 Update Bearer Response to the SGW-service pod with the cause as Request-Accepted.
7	The SGW-service pod requests the Usage report query when charging enabled with QoS trigger and QoS Change detected. The SGW-service pod sends the Sx Session Modification Request to the PFCP proto.
8	The SGW-service pod receives the Sx Session Modification Response from the PFCP proto with the cause as Request-Accepted.
9	The SGW-service pod triggers the CDR generation.
10	The SGW-service pod sends the CDR report to the GTPP-EP.
11	The GTPC-EP receives the Update Bearer response from the SGW-service pod with the cause as Request-Accepted.
12	The PGW forwards the S5 the Update Bearer response from the GTPC-EP with the cause as Request-Accepted.
13	The GTPP-EP sends the CDR ACK to the SGW-service pod.

Standards Compliance

The SGW Charging support complies with the following 3GPP standards:

- *3GPP TS 32.251 "Telecommunication management;Charging management;Packet Switched (PS) domain charging"*
- *3GPP TS 32.295 "Telecommunication management; Charging management; Charging Data Record (CDR) transfer"*
- *3GPP TS 32.297 "Telecommunication management; Charging management; Charging Data Record (CDR) file format and transfer"*
- *3GPP TS 32.298 "Telecommunication management; Charging management; Charging Data Record (CDR) parameter description"*

Limitations

- CN SGW does not support sending of CDR Records to CGF Server using GTPP
- The CGF server related functionalities including configuration, messaging, and redundancy are not supported.
- CN SGW cannot capture CDRs via the monitor protocol due to lack of GTPP support. Monitor Subscriber captures them using internal messages between **`sgw-service`** and **`gtpp-ep`** pods
- CN SGW does not support SGW specific CDR triggers for **`rat-change`** and **`apn-ambr-change`**

- CN SGW does not support CDR attribute **Served PDP/PDN Address Extension** for dual stack (IPv4v6) calls
- Supports only one CDR Dictionary: custom24
- Supports only three custom file formats: custom1(default) , custom5 and custom3
- CN SGW configuration currently supports only **`apn-ambr`** and **`node-id-prefix`** CDR attributes to be enabled or disabled. Rest all SGW CDR attributes are enabled by default.

Feature Configuration

Configuring this feature involves the following steps:

- CLI Configuration-This configuration provides commands to configure cnSGW-C charging profile, mode, threshold, and its characteristics. For more information, refer to CLI Configuration, on page 630.
- Show CLI-This configuration provides the commands to display the SFTP push CLI. For more information, refer to Show CLI, on page 641.

CLI Configuration

cnSGW-C charging CLI configuration involves the following steps:

- Charging Profile or GTP Prime-This configuration provides commands to configure cnSGW-C GTPP profile. For more information, refer to Configuring the cnSGW-C Charging Profile or GTP Prime, on page 631.
- Charging Mode-This configuration provides commands to configure the cnSGW-C charging mode. For more information, refer to Configuring the Charging Mode, on page 636.
- Charging Threshold-This configuration provides commands to configure the cnSGW-C charging threshold. For more information, refer to Configuring the cnSGW-C Charging Threshold, on page 636.
- Charging Threshold and Charging Profile Association-This configuration provides commands to configure cnSGW-C charging threshold and cnSGW-C charging profile association. For more information, refer to Configuring cnSGW-C Charging Threshold and cnSGW-C Charging Profile Association, on page 638.
- Call Control Profile-This configuration provides commands to configure cnSGW-Ccall control profile. For more information, refer to Configuring Call Control Profile, on page 639.
- Charging Characteristics Under Call Control Profile-This configuration provides commands to configure cnSGW-C charging characteristics under call control profile. For more information, refer to Configuring Charging Characteristics Under Call Control Profile, on page 640.

Configuring the cnSGW-C Charging Profile or GTP Prime

Note

- cnSGW-C charging supports multiple replicas of GTP Prime.
- cnSGW-C switches from primary storage server to secondary storage server on four consecutive failures with the primary storage server. It switches back to primary storage server on four consecutive failures to secondary storage server or after 30 minutes of switchover from primary storage server to secondary storage server whichever is earlier.
- When CDR file storage reaches beyond 95% of its allocated size, then old CDR files are deleted.

Configuring cnSGW-C charging profile or GTP prime involves the following steps:

- GTPP profile-This configuration provides commands to configure cnSGW-C GTPP profile. For more information, refer to Configuring the GTPP Profile, on page 631.
- Existing endpoint-related CLI-This configuration provides commands to configure cnSGW-C existing endpoint-related CLI. For more information, refer to Configuring the GTPP Endpoint, on page 634.
- SGW charging profile--This configuration provides commands to configure cnSGW-C GTPP profile. For more information, refer to Configuring SGW Charging Profile, on page 634.

Configuring the GTPP Profile

You can configure server details, dictionary, timeout, and so on, to use by the GTPP-EP pod.

To configure the GTPP profile, use the following configuration:

```
config
  profile gtpp-profile profile_name gtpp
    local-storage
      file
        rotation
         volume volume_value
         cdr-count cdrcount_value
         time-interval interval_value
        exit
        name
         prefix prefix_value
         format format
         max-file-seq-num max_sequence_number
         start-file-seq-num start_sequence_number
         recover-file-seq-num { true | false }
        exit
        purge-processed-files purge-interval purgeinterval_value
       exit
       push
        encrypted-url url_name
        encrypted-secondary-url url_name
       exit
      exit
```

```
        dictionary  24
        end
```

NOTES:

- **local-storage**—Local storage details.
- **file**—Specify the file details.
- **rotation**—Specify the file rotation details.
- **volume** *volume_value*—Specify the file volume in MiB for file rotation. Must be an integer in the range of 2-40. Default value is 4.
- **cdr-count** *cdrcount_value*—Specify the CDR count for file rotation. Must be an integer in the range of 1000-65000. Default value is 10000.
- **time-interval** *interval_value*—Specify the time interval in seconds for file rotation. Must be an integer in the range of 30-86400. Default value is 3600.
- **prefix** *prefix_value*—Specify the file name prefix to be used. If the prefix value isn't specified, the configuration takes default profile name.
- **format** *format*—Specify the file name format to be used to override the name format associated with the file format.
- **max-file-seq-num** *max_sequence_number*—Specify the maximum file sequence number to rollover. Default value is 4294967295.
- **start-file-seq-num** *start_sequence_number*—Specify the start sequence number during rollover. Default value is 1.
- **recover-file-seq-num { true | false }**—When set to true, file sequence number continues from the last sequence number on application restart. Default value is false.
- **purge-processed-files** —Enables periodic purging of processed files.
- **purge-processed-files purge-interval** *purgeinterval_value*—Specify the purging interval of processed files in minutes. Default value is 60.
- **encrypted-url**—Specify the primary SFTP URL to push CDR files to.
- **encrypted-secondary-url**—Specify the secondary SFTP URL to push when push fails on primary host.
- **dictionary** *custom_value*—Specify the dictionary to be used to ASN.1 encode a CDR.

Note

- The path in SFTP URL is by default a relative path to home directory of SFTP URL user specified in URL.

  Example: `encrypted-url sftp://user:pass@example.com:2020/upload/pf1`. It pushes files to `%USER_HOME/upload/pf1`

  Example: `encrypted-url sftp://user:pass@example.com:2020`. It pushes files to `%USER_HOME`

- To upload files to a folder outside the user's home directory, configure an absolute path by preceding the path with // at the beginning of the SFTP server path.

  Example: `encrypted-url sftp://user:pass@example.com:2020//var/opt`. It pushes the files to absolute path `/var/opt`

  SFTP user must have the write access to this path for the upload to be successful.

If password contains any special character outside the permissible URL character set, they must be percent coded as per the RFC 3986. For example, a URL with password `pass!word`, entered as `sftp://user:pass%21word@example.com/path/to/folder`

Configuration Example

The following is an example configuration.

```
config
   profile gtpp-profile pf1 gtpp
      local-storage
         file
            rotation
               volume   5
               cdr-count 1000
               time-interval 60
               exit
               name
                  prefix  NYPCF508
                  format  .%Y-%m-%d%H-%M-%S.%4Q
                  max-file-seq-num     4
                  start-file-seq-num   1
                  recover-file-seq-num false
                  exit
               purge-processed-files purge-interval 10
               exit
               push
                  encrypted-url sample.com sftp://user:pass@example.com//var/opt
                  encrypted-secondary-url sftp://user:pass@mirror.example.com//var/opt
                  exit
               exit
               dictionary custom24
               end
```

Configuring the GTPP Endpoint

Note

- GTPP-EP pod uses this configuration.
- GTPP-EP pod always ignores nodes configuration.
- When **k8s single-node** is set to **`false`**, it spawns two replicas of GTPP-EP pod in active or standby mode independent of replicas and nodes configuration.
- When **k8s single-node** is set to **`true`**, the configured replicas have its impact.
- When **k8s use-volume-claim** is set to **`true`**, endpoint GTP prime is used to set the storage size limit. Default value of storage size limit is one GB.
- When system is up and running, we can't change the storage size.

To configure GTPP endpoint, use the following commands:

```
config
 instance instance-id instance_id
  endpoint gtpprime
     replicas replicas_count
     nodes nodes_count
     storage storage_capacity
     end
```

NOTES:

- **replicas** *replicas_count*—Specify the number of replicas per node. Must be an integer.
- **nodes** *nodes_count*—This property is ignored. You may skip configuring it.
- **storage** *storage_capacity*—Specify the storage size of persistent volume in GB. Must be an integer in the range of 1-20.

Note CLI doesn't allow changing storage size while system is running. To change the storage size, bring the system down first.

Configuration Example

The following is an example configuration.

```
config
   instance instance-id 1
      endpoint gtpprime
         replicas 1
         storage  2
         end
```

Configuring SGW Charging Profile

This section describes how to configure SGW Charging profile.

You can configure the SGW charging profile for the following:

- Attribute details and adding them to the CDRs
- Different triggers in generating CDR

SGW service pod uses this configuration in cnSGW-C charging.

Use following commands to configure cnSGW-C charging profile.

```
config
  profile sgw-charging-profile profile_name
    gtpp-triggers
      volume-limit { enable | disable }
      time-limit { enable | disable }
      serving-node-change-limit { enable | disable }
      serving-node-plmn-change { enable | disable }
      uli-change { enable | disable }
      qos-change { enable | disable }
      ms-timezone-change { enable | disable }
    gtpp-attributes
      apn-ambr
        include-for-all-bearers
        include-for-default-bearer
        include-for-non-gbr-bearers
      node-id-suffix suffix_value
    gtpp-profile association_profile_name
    exit
```



Note The value of node-id-suffix is implementation-specific. However, it's recommended to give same value as prefix configured as a part of GTPP Profile.

NOTES:

- **apn-ambr**—Includes APN-AMBR value in CDR.
- **node-id-suffix** *suffix_value*—Specify the node ID suffix to include in NodeId field of CDR.
- **ms-timezone-change { enable | disable }**—Specify enable or disable the MS time zone change as a trigger for CDR generation. Default value is enable.
- **qos-change { enable | disable }**—Specify enable or disable the QoS change as a trigger for container addition to CDR. Default value is enable.
- **serving-node-change-limit { enable | disable }**—Specify enable or disable the serving node change (address) as a trigger for CDR generation. Default value is enable.
- **serving-node-plmn-change { enable | disable }**—Specify enable or disable the serving node PLMN change as a trigger for CDR generation.
- **time-limit { enable | disable }]**—Specify enable or disable the time limit breach as a trigger for CDR generation. Default value is enable.

- **uli-change { enable | disable }**—Specify enable or disable the ULI change as a trigger for container addition to CDR. Default value is enable.
- **volume-limit { enable | disable }**—Specify enable or disable the volume limit breach as a trigger for CDR generation. Default value is enable and that is included in NodeId field of CDR.

Configuration Example

The following is an example configuration.

```
config
   profile sgw-charging-profile ch1
      gtpp-triggers volume-limit enable
      gtpp-triggers time-limit  enable
      gtpp-triggers serving-node-change-limit disable
      gtpp-triggers uli-change  enable
      gtpp-triggers qos-change  disable
      gtpp-triggers ms-timezone-change disable
      gtpp-attributes apn-ambr include-for-all-bearers
      gtpp-attributes node-id-suffix test
      gtpp-profile pf1
      end
```

Configuring the Charging Mode

Charging mode configures the cnSGW-C service mode for accounting GTPP or none (default).

Note Enable offline charging when charging mode is set to GTPP.

To configure charging mode, use the following configuration:

```
config
   profile sgw sgw_srv_name
      charging-mode { gtpp | none }
      sgw-charging-threshold sgw_threshold_name
      sgw-charging-profile  sgw_charging-profile_name
      end
```

NOTES:

- **charging-mode { gtpp | none }**—Specify cnSGW-C charging mode.
- **sgw-charging-threshold** *sgw_threshold_name*—Specify the name of associated cnSGW-C charging threshold
- **sgw-charging-profile** *sgw_charging-profile_name*—Specify the name of associated cnSGW-C charging profile

Configuring the cnSGW-C Charging Threshold

cnSGW-C charging threshold configuration helps in configuring the thresholds or limits corresponding to volume or duration or buckets per CC (charging-characteristics).

Configuration of cnSGW-c charging threshold can be done in two ways.

Method - 1

```
config
  profile sgw-charging-threshold threshold_name
    cc profile value cc_profile_value
      volume total total_value
      buckets buckets_value
      duration duration_value
      end
```

Method - 2

```
config
  profile sgw-charging-threshold threshold_name
    cc profile value cc_profile_value
      volume
        total total_value
        uplink uplink_value
        downlink downlink_value
      volume total
      buckets buckets_value
      serving-node-changesnode_changes_value
      duration duration_value
      end
```

NOTES:

- **buckets** *buckets_value*—Specify the number of traffic volume container changes due to QoS change or other triggers before an accounting record must be closed. It ranges 1–20 and the default value is 4.
- **duration** *duration_value*—Specify the normal time duration that must elapse before closing an accounting record.
- **volume total**—Specify the CC volume details.

Configuration Example

The following is an example configuration:

```
config
   profile sgw-charging-threshold thre1
      cc profile value 1
         volume total 100000
         buckets  1
         duration 60
         end

config
   profile sgw-charging-threshold thre1
      cc profile value 2
         volume uplink 100000
         volume downlink 100000
         buckets  1
         serving-node-changes 4
         duration 120
         end
```

Note When **gtpp-triggers serving-node-change-limit** is enabled and **serving-node-changes** configured under SGW charging threshold, CDR gets generated after 4 times serving node changes (MME).

Configuring cnSGW-C Charging Threshold and cnSGW-C Charging Profile Association

This section describes how to configure the SGW Charging Threshold and SGW Charging profile association.

This configuration associate **sgw-charging-threshold** and **sgw-charging-profiles** to the SGW profile.

Configuration of cnSGW-c charging threshold and cnSGW-c charging profile association can be done in two ways.

Method - 1

To configure cnSGW-c charging threshold and cnSGW-c charging profile association, use the following commands.

```
config
  profile sgw sgw_srv_name
    locality location_code
    fqdn dnn_name
    plmn-id
      mcc mcc_value
      mnc mnc_value
    charging-mode { gtpp | none }
    sgw-charging-profile value
    sgw-charging-threshold limit_name
    end
```

Method - 2

Use the following commands to configure SGW Charging Threshold and SGW Charging Profile association.

```
config
  profile sgw sgw_srv_name
    sgw-charging-threshold threshold_value
    locality location_code
    fqdn dnn_name
    charging-mode mode_name
    subscriber-policy policy_name
    end
```

Configuration Example

The following is an example configuration.

```
config
  profile sgw sgw1
    locality LOC1
    fqdn 209.165.200.254
    allowed-nssai [ slice1 ]
    plmn-id mcc 123
    plmn-id mnc 456
```

```
      charging-mode gtpp
      sgw-charging-profile ch1
      sgw-charging-threshold limit1
      end
config
   profile sgw sgw1
      sgw-charging-threshold thre1
      locality  LOC1
      fqdn 209.165.200.254
      charging-mode none
      subscriber-policy polSub
      end
```

Configuring Call Control Profile

Call control profile configuration defines and applies the call handling rules through an operator policy.

The charging mode value from the call control profile overrides the configured value in cnSGW-C profile.

Note

- One call control profile is associated with one operator policy
- It's a standalone configuration

Configuring cnSGW-C call control profile involves the following steps:

- Call Control Profile Creation-This configuration provides commands to configure cnSGW-C call control profile Creation. For more information, refer to Configuring the Call Control Profile Creation, on page 639.
- Operator Policy Association-This configuration provides commands to configure cnSGW-operator policy association. For more information, refer to Configuring the Operator Policy Association, on page 639.

Configuring the Call Control Profile Creation

To configure the call control profile creation, use the following configuration:

```
config
   policy call-control-profile call_control_profile_name
      charging-mode sgw_charging_mode
      sgw-charging-profile assocaited_sgw_charging_profile
      end
```

Configuration Example

The following is an example configuration.

```
config
   policy call-control-profile ccp1
      charging-mode gtpp
      sgw-charging-profile ch1
      end
```

Configuring the Operator Policy Association

To configure the operator policy association, use following configuration:

```
config
   policy operator operator_name
   policy dnn  dnn_policy_name
   policy network-capability network_name
   call-control-profile value
   end
```

Configuration Example

The following is an example configuration.

```
config
   policy operator opPol1
   policy dnn polDnn
   policy network-capability nc1
   call-control-profile ccp1
   end
```

Configuring Charging Characteristics Under Call Control Profile

You can define local values and select the source of charging characteristics for charging decisions.

To configure charging characteristics under call control profile, use the following configuration:

```
config
   policy call-control-profile  call_control_profile_name
      sgw-charging-profile charging_type
      charging-mode mode_type
      cc prefer preference_type
      cc local-value profile index_bit
      end
```

NOTES:

- **cc prefer local-value** and **cc prefer hlr-hss-value** are optional parameters.
- **cc prefer { hlr-hss-value | local-value }**—Specify a preference to use in charging characteristics from the following:
  - When received from HLR or HSS through MME and preference set to hlr-hss.
  - When preference set to local-value. See the following CLI:

    ```
    cc prefer local-value
    cc local-value profile index-bit
    ```

- **cc local-value profile** —Specify the local-value parameter information as follows:
  - index_bit default value is 8
  - Sets the local value of the profile index for the charging characteristics, when the charging characteristics(CC) prefer value is set to local-value

Configuration Example

The following is an example configuration.

```
config
   policy call-control-profile CCP
```

```
      sgw-charging-profile test
      charging-mode gtpp
      cc prefer local-value
      cc local-value profile 4
      end
config
   policy call-control-profile CCP1
      sgw-charging-profile test
      charging-mode gtpp
      cc prefer hlr-hss-value
      end
```



Note Use the system default configured value as 8 otherwise use the value which comes in CSR.

```
config
   policy call-control-profile CCP2
      sgw-charging-profile test
      charging-mode gtpp
      cc prefer local-value
      end
```



Note Default value for cc profile is 8.

Show CLI

GTPP-EP SFTP Push CLI

- **show gtpp-ep endpoints**: Displays the list of running GTPP-EP pods and their corresponding IPs
- **show gtpp-ep files endpoint** *pod-name* **profile** *gtpp-profile_name*: Displays the archived files on specific GTPP-EP pod for the given gtpp
- **cdr push endpoint** *pod-name* **profile** *gtpp-profile* **filename** *file-to-be-uploaded*: Pushes the available file to archive folder on specific GTPP-EP pod for given GTPP profile.

CDR Fields Supported in cnSGW-CDRs

The tables in this section list the cnSGW-CDR fields present in the available dictionaries.

custom24 Dictionary

Table 229: custom24 Dictionary Description

Field Name	Tag Number	Category	Description
Record Type	0	M	SGW IP-CAN bearer record.

Field Name	Tag Number	Category	Description
Served IMSI	3	M	IMSI of the served party.
S-GW Address	4	M	The control plane IP address of the SGW used.
S-GW BINARY IPV4 ADDRESS	4-0	M	The octet string includes the Gn address of the GGSN service in binary coding.
S-GW BINARY IPV6 ADDRESS	4-0	M	The octet string included in the field described includes the Gn address of the GGSN service in binary coding.
Charging ID	5	M	IP-CAN bearer identifier. To identify IP-CAN bearers created by PCNs in different records
List of Serving Node Address	6	M	List of serving node control plane IP addresses (Example: SGSN, MME) used during this record.
Serving Node BINARY IPV4 ADDRESS	6-0	M	The octet string included in the field described above includes the IPv4 address of the MME.
Serving Node BINARY IPV6 ADDRESS	6-0	M	The octet string included in the Serving node binary IPv4 address field includes the IPV6 address of the MME.
Access point name network identifier	7	M	The logical name of the connected access point to the external packe data network (network identifier part of APN).
PDP/PDN Type	8	M	This field indicates PDN type (Example IPv4, IPv6 or IPv4v6).
Served PDP/PDN Address	9	M	IP address allocated for the PDP context or PDN connection, if available. IPv4 when PDN type is IPv4 or IPv6 when PDN type is IPv6 or IPv4v6.
PDP IP Address	9-0	M	This field contains the IP address for the PDP context.
PDP IPv4 Address	9-0-0	M	The octet string included in the PDP IP address field includes the th SGW assigned IPv4 address to the subscriber in binary format.
PDP IPv6 Address	9-0-0	M	The octet string included in the PDP IP address field includes the IPv6 address assigned to the subscriber by the SGW in binary coding
Dynamic Address Flag	11	O	Indicates whether served PDP/PDN address is dynamic, which is allocated during IP-CAN bearer activation, initial attach (E-UTRAN or over S2x) and UE requested PDN connectivity. This field is missing if address is static.

Field Name	Tag Number	Category	Description
List of Traffic Data Volumes	12	M	A list of changes in charging conditions for QCI, ARP pair, eac change is time stamped. Charging conditions categorize traffic volumes, such as per tariff period. Initial and subsequently char QoS and corresponding data values are also listed.
Change of charging condition	12-0	M	Each traffic volume container contains details of a charging condi A new container is usually created for a QoS change and for tar changes.
Data Volume GPRS Uplink	12-0-3	M	This field is a part of the ChangeOf CharCondition element in the List of Traffic Volumes. It includes the number o octets received in the uplink direction during the timeframe speci by the container. For each new container, the counter is reset an does not accumulate.
Data Volume GPRS Downlink	12-0-4	M	This field is a part of the ChangeOf CharCondition element in the List of Traffic Volumes. It includes the number o octets transmitted in the downlink direction during the timefram specified by the container. For each new container, the counter reset and does not accumulate.
Change Condition	12-0-5	M	This field is part of the ChangeOf CharCondition element in the of Traffic Volumes. It defines the change in user plane to UE.
Change Time	12-0-6	M	This field is part of the ChangeOf CharCondition element in the of Traffic Volumes. It provides the local time when a change condition (example: record closure) occurred and the container closed.
User Location Information	12-0-8	O	This field contains the User Location Information.
EPC QoS Information	12-0-9	O	In case of IP-CAN bearer specific container, this field contains authorized QoS for the IP-CAN bearer. First container for each QCI/ARP pair includes this field. In the following containers th field is present if previous change condition is "QoS change". T field is applicable only in SGW-CDR.
CP CIoT EPS Optimisation Indicator	12-0-19	O	The cPCIo TEPSOptimisation Indicator field indicates whether Control Plane CIoT EPS optimisation is used for the transfer of data volume captured by the container. This is included in the Tr data container only if previous container's change condition is "cha in user plane to UE". Note, the CP CIoT EPS Optimisation indic field in SGW-CDR main level contains the CP CIoT EPS optimisa indicator value when SGW-CDR was opened.

Field Name	Tag Number	Category	Description
QCI	12-9-1	M	—
Uplink MBR	12-9-2	O	—
Down link MBR	12-9-3	O	—
Uplink GBR	12-9-4	O	—
Down link GBR	12-9-5	O	—
arp	12-9-6	O	—
APN AMBR Uplink	12-9-7	O	—
APN AMBR Downlink	12-9-8	O	—
Extended Maximum Requested BW UL	12-9-9	O	—
Extended Maximum Requested BW DL	12-9-10	O	—
Extended GBR UL	12-9-11	O	—
extended GBRDL	12-9-12	0	—
Extended APN AMBR UL	12-9-13	O	—
Extended APN AMBR DL	12-9-14	O	—
Record Opening Time	13	M	Time stamp when IP-CAN bearer is activated in this S-GW or record opening time on subsequent partial records.
Duration	14	M	This field contains the duration in seconds for the record.
Cause for Record Closing	15	M	This field contains a reason for the closure of the CDR.
Diagnostics	16	O	This field is included in the CDR when the bearer context is released and when the **gtpp attribute diagnostics** is configured.
gsm408cause	16-0	M	—
Record Sequence Number	17	O	Partial record sequence number, only present in case of partial records
Node ID	18	O	Name of the recording entity.

Field Name	Tag Number	Category	Description
Record Extensions	19	O	A set of network operator or manufacturer specific extensions to record. Conditioned when the extension is available.
Local Record Sequence Number	20	O	Consecutive record number created by this node. The number is allocated sequentially including all CDR types.
APN Selection Mode	21	M	An index indicating how the APN is selected.
Served MSISDN	22	M	The primary MSISDN of the subscriber.
Charging Characteristics	23	M	The charging characteristics that are applied to the IP-CAN bea
Charging Characteristics Selection Mode	24	O	Holds the information about how charging characteristics are sele
IMS Signaling Context	25	O	Included if the IM-CN Subsystem Signalling Flag is set, see [2 IP-CAN bearer is used for IMS signalling.
Serving Node PLMN Identifier	27	O	Serving node PLMN Identifier (MCC and MNC) used during th record, if available.
Served IMEISV	29	O	IMEISV of the ME, if available.
RAT Type	30	O	This field indicates the Radio Access Technology (RAT) type currently used by the Mobile Station, when available.
MS Time Zone	31	O	The Time Zone IE that the MME may provide to the SGW duri the PDN context activation or modification procedure.
User Location Information	32	O	This field contains the user location information as described in 29.274 for eGTP case (Example: CGI, SAI, RAI TAI and ECG This field is provided by the SGSN or MME and transferred to SGW or PGW during the IP-CAN bearer activation or modifica
S-GW Change	34	O	This field is present only in the SGW-CDR to indicate that this i first record after an SGW change. In this case, it is set to TRUE (
Serving Node Type	35	M	These fields contain one or several serving node types in contro plane of SGW or PGW, which is connected during the record. T serving node types listed here map to the serving node addresse listed in the field Serving node Address in sequence.
Serving Node Type enum	35-1	M	—
P-GW Address Used	36	M	This field is the PGW IP address for the control plane.

Field Name	Tag Number	Category	Description
P-GW Binary IPV4 Address	36-0	M	This field includes the PGW assigned IPv4 address to the subscribe in binary format.
P-GW Binary IPV6 Address	36-0	M	This field includes the PGW assigned IPv6 address to the subscribe in binary format.
P-GW PLMN Identifier	37	O	—
Start Time	38	O	This field holds the time when User IP-CAN session starts. It's available in the CDR for the first bearer in an IP-CAN session.
Stop Time	39	O	This field holds the time when User IP-CAN session is terminated. It's available in the CDR for the last bearer in an IP-CAN session.
PDN Connection ID	40	O	This field holds the PDN connection (IP-CAN session) identifier to identify different records belonging to same PDN connection.
iMSI unauthenticated Flag	41	O	This field indicates the provided served IMSI is not authenticated (emergency bearer service situation).
user CSG Information	42	O	This field contains the User CSG Information status of the user accessing a CSG cell. It comprises CSG ID within the PLMN, Acces mode and indication on CSG membership for the user when hybrid access applies, as defined in *TS 29.060* for GPRS case, and in *TS 29.274* for EPC case.
cSGId	42-0	O	A CSG ID is a unique identifier within the scope of PLMN which identifies a Closed Subscriber Group (CSG) in the PLMN associated with a CSG cell or group of CSG cells.
cSGAccess Mode	42-1	O	cSGAccessMode. It's either closed or hybrid.
cSG Membership Indication	42-2	O	This field provides an indication on CSG membership for the user.
Served PDP PDN Address Extension	43	O	This field contains the IPv4 address for the PDN connection (PDP context, IP-CAN bearer) when dual-stack IPv4 IPv6 is used, and the IPv6 adress is included in served PDP address or served PDP or PDN address.
PDP IP Address	43-0	M	This field contains the IP address for the PDP context.
PDP IPV4 Address	43-0-0	M	This field includes the IPv4 address assigned to the subscriber by SGW in binary coding.
lowAccess Priority Indicator	44	O	This field indicates if the PDN connection has a low priority, which is for machine type communication.

Field Name	Tag Number	Category	Description
dynamic Address FlagExt	47	O	This field indicates whether served IPv4 PDP or PDN address i dynamic, which is allocated during IP-CAN bearer activation, ir attach (E-UTRAN or over S2x) and UE requested PDN connecti with PDP or PDN type IPv4v6. This field is missing if IPv4 add is static.
s-GW iPv6 Address	48	O	The control plane IPv6 address, in case of IPv4v6 dual stack, o S-GW.
SGW BINARY IPV6 ADDRESS	48-0	O	This field includes the Gn address of the GGSN service in bina format.
List of Serving Node IPv6Address	49	O	List of serving node control plane IPv6 addresses, in case of IPv dual stack, (Example: S4-SGSN, MME) used during this record
Serving Node BINARY IPV6 ADDRESS	49-0	M	The octet string in this field includes the IPV6 address of the M
p-GW iPv6 Address Used	50	O	This field is the PGW IPv6 Address, in case of IPv4v6 dual stac for the control plane.
PGW BINARY IPV6 ADDRESS	50-0	O	The octet string in this field includes the IPV6 address assigned the subscriber by of the P-GW in binary coding.
last User Location Information	55	O	Indicates the UE's last user location information during bearer deactivation or session release.
last MSTime Zone	56	O	Indicates the Latest timezone of UE while bearer deactivation o session release.
CP CIoT EPS Optimisation Indicator	59	O	This field indicates whether Control Plane CIoT EPS optimisat is used by the PDN connection during data transfer with the UE (is, Control Plane NAS PDU viaS11-U between S-GW and MME not (that is, User Plane via S1-U between S-GW and eNB).
UNI PDU CP Only Flag	60	O	The uNIPDUCP OnlyFlag field indicates whether this PDN connection is applied "Control Plane Only flag", that is, transferred using Control Pla NAS PDUs only, when Control Plane CIoT EPS Optimisation i enabled. This field is not flagged when both user plane and con plane UNI for PDU transfer (that is, S1-U and S11-U from S-G are allowed, when Control Plane CIoT EPS Optimisation is enab
List of RAN Secondary RAT Usage Reports	64	OC	This field includes one or more containers reported from the RA for a secondary RAT.
RAN Secondary RAT Usage Report	64-0	M	This field includes RAN reported containers for a secondary RA

Field Name	Tag Number	Category	Description
Data Volume Uplink	64-0-1	M	This field includes the number of octets transmitted during the usage of the packet data services in the uplink direction reported from RAN. The counting and reporting from RAN of uplink data volumes is optional.
Data Volume Downlink	64-0-2	M	This field includes the number of octets transmitted during the usage of the packet data services in the downlink direction reported from RAN. The counting and reporting from RAN of downlink data volumes is optional.
RAN Start Time	64-0-3	M	This field is a timestamp at which RAN opens the volume container
RAN End Time	64-0-4	M	This field is a time stamp at which RAN closes the volume container
Secondary RAT Type	64-0-5	OC	This field contains the RAT type for the secondary RAT.
UE Local IP Port Info	253	O	This field includes the S2b user local IP port information.
UE Local IP Address	253-0	O	This field includes the UWAN user IP address.
UDP Source Port	253-1	O	This field includes the UWAN user source port.

Note All IP addresses are encoded in binary format.

ASN.1 Definition for Fields in custom24

The following section provides the complete ASN.1 definition of all cnSGW-CDR related fields in the custom24 dictionary.

```
GPRS-SGW-Charging-DataTypes-REL8 DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-------------------------------------------------------------------------------
--
--    GPRS RECORDS
--
-------------------------------------------------------------------------------

GPRSRecord::= CHOICE
--
-- Record values 20, 22..27 are specific
-- Record values 76..77 are MBMS specific
-- Record values 78..79 are EPC specific
{
    sGWRecord[78] SGWRecord
}

SGWRecord    ::= SET
{
    recordType                                  [0] RecordType,
    servedIMSI                                  [3] IMSI,
    s-GWAddress                                 [4] GSNAddress,
```

```
    chargingID                                  [5] ChargingID,
    servingNodeAddress                          [6] SEQUENCE OF GSNAddress,
    accessPointNameNI                           [7] AccessPointNameNI OPTIONAL,
    pdpPDNType                                  [8] PDPType OPTIONAL,
    servedPDPPDNAddress                         [9] PDPAddress OPTIONAL,
    dynamicAddressFlag                          [11] DynamicAddressFlag OPTIONAL,
    listOfTrafficVolumes                        [12] SEQUENCE OF ChangeOfCharCondition
OPTIONAL,
    recordOpeningTime                           [13] TimeStamp,
    duration                                    [14] CallDuration,
    causeForRecClosing                          [15] CauseForRecClosing,
    diagnostics                                 [16] Diagnostics OPTIONAL,
    recordSequenceNumber                        [17] INTEGER OPTIONAL,
    nodeID                                      [18] NodeID OPTIONAL,
    recordExtensions                            [19] ManagementExtensions OPTIONAL,
    localSequenceNumber                         [20] LocalSequenceNumber OPTIONAL,
    apnSelectionMode                            [21] APNSelectionMode OPTIONAL,
    servedMSISDN                                [22] MSISDN OPTIONAL,
    chargingCharacteristics                     [23] ChargingCharacteristics,
    chChSelectionMode                           [24] ChChSelectionMode OPTIONAL,
    iMSsignalingContext                         [25] NULL OPTIONAL,
    servingNodePLMNIdentifier                   [27] PLMN-Id OPTIONAL,
    servedIMEISV                                [29] IMEI OPTIONAL,
    rATType                                     [30] RATType OPTIONAL,
    mSTimeZone                                  [31] MSTimeZone OPTIONAL,
    userLocationInformation                     [32] OCTET STRING OPTIONAL,
    sGWChange                                   [34] SGWChange OPTIONAL,
    servingNodeType                             [35] SEQUENCE OF ServingNodeType,
    p-GWAddressUsed                             [36] GSNAddress OPTIONAL,
    p-GWPLMNIdentifier                          [37] PLMN-Id OPTIONAL,
    startTime                                   [38] TimeStamp OPTIONAL,
    stopTime                                    [39] TimeStamp OPTIONAL,
    pDNConnectionID                             [40] ChargingID OPTIONAL,
    servedPDPPDNAddressExt                      [43] PDPAddress OPTIONAL,
    lowAccessPriorityIndicator                  [44] NULL OPTIONAL,
    dynamicAddressFlagExt                       [47] DynamicAddressFlag OPTIONAL,
    s-GWiPv6Address                             [48] GSNAddress OPTIONAL,
    servingNodeiPv6Address                      [49] SEQUENCE OF GSNAddress OPTIONAL,
    p-GWiPv6AddressUsed                         [50] GSNAddress OPTIONAL,
    lastUserLocationInformation                 [55] OCTET STRING OPTIONAL,
    lastMSTimeZone                              [56] MSTimeZone OPTIONAL,
    cPCIoTEPSOptimisationIndicator              [59] BOOLEAN OPTIONAL,
    uNIPDUCPOnlyFlag                            [60] BOOLEAN OPTIONAL,
    listOfRANSecondaryRATUsageReports           [64] SEQUENCE OF RANSecondaryRATUsageReport
 OPTIONAL,
    uELocalIPAddressPort                       [253] SEQUENCE OF UELocalIPPortInfo OPTIONAL

}

AccessPointNameNI::= IA5String (SIZE(1..63))
 --
 -- Network Identifier part of APN in dot representation.
 -- For example, if the complete APN is 'apn1a.apn1b.apn1c.mnc022.mcc111.gprs'
 -- NI is 'apn1a.apn1b.apn1c' and is presented in this form in the CDR.

APNSelectionMode::= ENUMERATED
{
 --
 -- See Information Elements TS 29.060, TS 29.274    or TS 29.275
 --
     mSorNetworkProvidedSubscriptionVerified    (0),
     mSProvidedSubscriptionNotVerified          (1),
     networkProvidedSubscriptionNotVerified     (2)
}
```

```
CallDuration::= INTEGER
                --
                -- The call duration is counted in seconds.
                -- For successful calls /sessions / PDP contexts, this is the chargeable
duration.
                -- For call attempts this is the call holding time.
                --


CauseForRecClosing ::= INTEGER
{
 --
 -- In PGW-CDR and SGW-CDR the value servingNodeChange is used for partial record
 -- generation due to Serving Node Address list Overflow
 -- In SGSN servingNodeChange indicates the SGSN change
 --
 -- LCS related causes belong to the MAP error causes acc. TS 29.002
 --
 -- cause codes 0 to 15 are defined 'CauseForTerm' (cause for termination)
 -- All cause values are not relevent to SGW. Refer the spec to find out the
          -- cause values for SGW.
     normalRelease                           (0),
     abnormalRelease                         (4),
     cAMELInitCallRelease                    (5),
     volumeLimit                             (16),
     timeLimit                               (17),
     servingNodeChange                       (18),
     maxChangeCond                           (19),
     managementIntervention                  (20),
     intraSGSNIntersystemChange              (21),
     rATChange                               (22),
     mSTimeZoneChange                        (23),
     sGSNPLMNIDChange                        (24),
     unauthorizedRequestingNetwork           (52),
     unauthorizedLCSClient                   (53),
     positionMethodFailure                   (54),
     unknownOrUnreachableLCSClient           (58),
     listofDownstreamNodeChange              (59)
}


ChangeCondition ::= ENUMERATED
{
     qoSChange                              (0),
     tariffTime                             (1),
     recordClosure                          (2),
     cGI-SAICHange                          (6),    -- bearer modification. CGI-SAI Change
     rAIChange                              (7),    -- bearer modification. RAI Change
     dT-Establishment                       (8),
     dT-Removal                             (9),
     eCGIChange                             (10),    -- bearer modification. ECGI Change
     tAIChange                              (11),    -- bearer modification. TAI Change
     apnAmbrChange                          (50)     -- apn-ambr change

}


ChangeOfCharCondition ::= SEQUENCE
{
        --
        -- qosRequested and qosNegotiated are used in S-CDR only
        -- ePCQoSInformation used in SGW-CDR,PGW-CDR, IPE-CDR, TWAG-CDR and ePDG-CDR only
        -- userLocationInformation is used only in S-CDR, SGW-CDR and PGW-CDR
```

```
        -- chargingID used in PGW-CDR only when Charging per IP-CAN session is active
        -- accessAvailabilityChangeReason and relatedChangeOfCharCondition applicable only
 in PGW-CDR
        -- cPCIoTOptimisationIndicator is used in SGW-CDR only
        --
        qosRequested                    [1] QoSInformation OPTIONAL,
        qosNegotiated                   [2] QoSInformation OPTIONAL,
        dataVolumeGPRSUplink            [3] DataVolumeGPRS OPTIONAL,
        dataVolumeGPRSDownlink          [4] DataVolumeGPRS OPTIONAL,
        changeCondition                 [5] ChangeCondition,
        changeTime                      [6] TimeStamp,
        userLocationInformation         [8] OCTET STRING OPTIONAL,
        ePCQoSInformation               [9] EPCQoSInformation OPTIONAL,
        chargingID                      [10] ChargingID OPTIONAL,
        userCSGInformation              [12] UserCSGInformation OPTIONAL,
        diagnostics                     [13] Diagnostics OPTIONAL,
        rATType                         [15] RATType OPTIONAL,
        uWANUserLocationInformation     [17] UWANUserLocationInfo OPTIONAL,
        cPCIoTEPSOptimisationIndicator [19] CPCIoTEPSOptimisationIndicator OPTIONAL


}

ChargingCharacteristics ::= OCTET STRING (SIZE(2))


ChargingID ::= INTEGER (0..4294967295)
 --
 -- Generated in P-GW, part of IP CAN bearer
 -- 0..4294967295 is equivalent to 0..2**32-1
 --

ChChSelectionMode    ::= ENUMERATED
{
     servingNodeSupplied        (0), -- For S-GW/P-GW
     subscriptionSpecific       (1), -- For SGSN only
     aPNSpecific                (2), -- For SGSN only
     homeDefault                (3), -- For SGSN, S-GW and P-GW
     roamingDefault             (4), -- For SGSN, S-GW and P-GW
     visitingDefault            (5) -- For SGSN, S-GW and P-GW
}

DataVolumeGPRS ::= INTEGER
 --
 -- The volume of data transferred in octets.
 --

DynamicAddressFlag ::= BOOLEAN


EPCQoSInformation ::= SEQUENCE
{
     --
     -- See TS 29.212 for more information
     --
     qCI                            [1] INTEGER,
     maxRequestedBandwithUL         [2] INTEGER OPTIONAL,
     maxRequestedBandwithDL         [3] INTEGER OPTIONAL,
     guaranteedBitrateUL            [4] INTEGER OPTIONAL,
     guaranteedBitrateDL            [5] INTEGER OPTIONAL,
     aRP                            [6] INTEGER OPTIONAL,
     apnAmbrUplink                  [7] INTEGER OPTIONAL,
     apnAmbrDownlink                [8] INTEGER OPTIONAL,
     extendedMaxRequestedBWUL       [9] INTEGER OPTIONAL,
```

```
     extendedMaxRequestedBWDL       [10] INTEGER OPTIONAL,
     extendedGBRUL                  [11] INTEGER OPTIONAL,
     extendedGBRDL                  [12] INTEGER OPTIONAL,
     extendedAPNAMBRUL              [13] INTEGER OPTIONAL ,
     extendedAPNAMBRDL              [14] INTEGER OPTIONAL
}

ETSIAddress::= AddressString
 --
 -- First octet for nature of address, and numbering plan indicator (3 for X.121)
 -- Other octets TBCD
 -- See TS 29.002
 --

GSNAddress::= IPAddress


MSNetworkCapability ::= OCTET STRING (SIZE(1..8))
 -- see TS 24.008

NetworkInitiatedPDPContext ::= BOOLEAN
 --
 -- Set to true if PDP context was initiated from network side
 --

NodeID ::= IA5String (SIZE(1..20))

NumberOfDPEncountered    ::= INTEGER


PDPAddress ::= CHOICE
{
     iPAddress         [0] IPAddress,
     eTSIAddress     [1] ETSIAddress
}

PDPType    ::= OCTET STRING (SIZE(2))
 --
 -- OCTET 1: PDP Type Organization
 -- OCTET 2: PDP Type Number
 -- See TS 29.060 for GTP, TS 29.274 for eGTP and TS 29.275 for PMIP
 --

PLMN-Id    ::= OCTET STRING (SIZE (3))
 --
 -- This is a 1:1 copy from the Routing Area Identity (RAI) IE specified in TS 29.060
 --            as follows:
 -- OCTET 1 of PLMN-Id = OCTET 2 of RAI
 -- OCTET 2 of PLMN-Id = OCTET 3 of RAI
 -- OCTET 3 of PLMN-Id = OCTET 4 of RAI
 --

QoSInformation ::= OCTET STRING (SIZE (4..255))
 --
 -- This    octet string
 -- is a 1:1 copy of the contents (i.e. starting with octet 5) of the "Bearer Quality of
 -- Service" information element specified in TS 29.274
 --

RANSecondaryRATUsageReport ::= SEQUENCE
                -- ]
{
                dataVolumeUplink              [1] DataVolumeGPRS,
                dataVolumeDownlink            [2] DataVolumeGPRS,
```

```
                rANStartTime                    [3] TimeStamp,
                rANEndTime                      [4] TimeStamp,
                secondaryRATType                [5] SecondaryRATType OPTIONAL
}

SecondaryRATType ::= INTEGER
{
                reserved (0),
                nR (1) -- New Radio 5G
}

RATType ::= INTEGER (0..255)
 --
 -- This integer is 1:1 copy of the RAT type value as defined in TS 29.060 for GTP,
 -- TS 29.274 for eGTP and TS 29.275 for PMIP.
 --

UWANUserLocationInfo ::= SEQUENCE
{
     uELocalIPAddress    [0] IPAddress,
     uDPSourcePort       [1] OCTET STRING (SIZE(2)) OPTIONAL,
     sSID               [2] OCTET STRING OPTIONAL,     -- see format in IEEE Std 802.11-2012
 [408]
     bSSID              [3] OCTET STRING OPTIONAL      -- see format in IEEE Std 802.11-2012
 [408]
}

RecordType    ::= INTEGER
{
    -- Record values 0..17 are CS specific.
    -- The contents are defined in TS 32.250

    sGWRecord         (84)
}

ResultCode ::= INTEGER
 -- charging protocol return value, range of 4 byte (0...4294967259)
 -- see Result-Code AVP as used in 3GPP 32.299
 --


ServingNodeType ::= ENUMERATED
{
     sGSN                 (0),
     pMIPSGW              (1),
     gTPSGW               (2),
     ePDG                 (3),
     hSGW                 (4),
     mME                  (5)
}

SGWChange ::= BOOLEAN
 --
 -- present if first record after inter S-GW change
 --

Diagnostics            ::= CHOICE
{
     gsm0408Cause                                    [0] INTEGER,
     -- See TS 24.008
     gsm0902MapErrorValue                    [1] INTEGER,
     -- Note: The value to be stored here corresponds to
     -- the local values defined in the MAP-Errors and
     -- MAP-DialogueInformation modules, for full details
```

```
     -- see TS 29.002
               itu-tQ767Cause                    [2] INTEGER,
     -- See ITU-T Q.767
               networkSpecificCause              [3] ManagementExtension,
     -- To be defined by network operator
               manufacturerSpecificCause     [4] ManagementExtension,
 -- To be defined by manufacturer
               positionMethodFailureCause [5] PositionMethodFailure-Diagnostic,
     -- see TS 29.002
               unauthorizedLCSClientCause [6] UnauthorizedLCSClient-Diagnostic
     -- see TS 29.002
}

IPAddress::= CHOICE
{
     iPBinaryAddress                         IPBinaryAddress,
     iPTextRepresentedAddress    IPTextRepresentedAddress
}

CPCIoTEPSOptimisationIndicator ::= BOOLEAN

IPBinaryAddress::= CHOICE
{
     iPBinV4Address     [0] OCTET STRING (SIZE(4)),
     iPBinV6Address     [1] OCTET STRING (SIZE(16))
}

IPTextRepresentedAddress::= CHOICE
{
 --
 -- IP address in the familiar "dot" notation
 --
     iPTextV4Address     [2] IA5String (SIZE(7..15)),
     iPTextV6Address     [3] IA5String (SIZE(15..45))
}

PositionMethodFailure-Diagnostic ::= ENUMERATED
{
                congestion                                              (0),
                insufficientResources                                   (1),
                insufficientMeasurementData                             (2),
                inconsistentMeasurementData                             (3),
                locationProcedureNotCompleted                           (4),
                locationProcedureNotSupportedByTargetMS                 (5),
                qoSNotAttainable                                        (6),
                positionMethodNotAvailableInNetwork                     (7),
                positionMethodNotAvailableInLocationArea                (8)
}

LocalSequenceNumber ::= INTEGER (0..4294967295)
 --
 -- Sequence number of the record in this node
 -- 0.. 4294967295 is equivalent to 0..2**32-1, unsigned integer in four octets


ManagementExtension ::= SEQUENCE
{
     identifier OBJECT IDENTIFIER,
     significance        [1]         BOOLEAN DEFAULT FALSE,
     information         [2]         ANY DEFINED BY identifier
}

ManagementExtensions ::= SET OF ManagementExtension
```

```
MSISDN::= ISDN-AddressString
                --
 -- See TS 23.003

MSTimeZone::= OCTET STRING (SIZE (2))
 --
 -- 1.Octet: Time Zone and 2. Octet: Daylight saving time, see TS 29.060


TimeStamp::= OCTET STRING (SIZE(9))
                --
                -- The contents of this field are a compact form of the UTCTime format
                -- containing local time plus an offset to universal time. Binary coded
                -- decimal encoding is employed for the digits to reduce the storage and
                -- transmission overhead
                -- e.g. YYMMDDhhmmssShhmm
                -- where
                -- YY     =            Year 00 to 99                    BCD encoded
                -- MM     =            Month 01 to 12                   BCD encoded
                -- DD     =            Day 01 to 31                       BCD encoded
                -- hh     =            hour 00 to 23                    BCD encoded
                -- mm     =            minute 00 to 59                BCD encoded
                -- ss     =            second 00 to 59                BCD encoded
                -- S      =            Sign 0 = "+", "-"             ASCII encoded
                -- hh     =            hour 00 to 23                    BCD encoded
                -- mm     =            minute 00 to 59                BCD encoded
                --

UELocalIPPortInfo   ::= SEQUENCE
{
    --
    -- The S2b user Local IP Port Information
    --
    uELocalIPAddress                          [0] IPAddress OPTIONAL,
    uDPSourcePort                              [1] INTEGER OPTIONAL
}

UELocalIPAddress::= IPAddress
UDPSourcePort           ::= INTEGER


UnauthorizedLCSClient-Diagnostic ::= ENUMERATED
{
                noAdditionalInformation                                    (0),
                clientNotInMSPrivacyExceptionList                          (1),
                callToClientNotSetup                                       (2),
                privacyOverrideNotApplicable                               (3),
                disallowedByLocalRegulatoryRequirements                    (4),
                unauthorizedPrivacyClass                                   (5),
                unauthorizedCallSessionUnrelatedExternalClient             (6),
                unauthorizedCallSessionRelatedExternalClient               (7)
}

CSGAccessMode ::= ENUMERATED
{
                 closedMode  (0),
                 hybridMode  (1)
}
CSGId         ::= OCTET STRING (SIZE(4))
                --
                -- Defined in 23.003. Coded according to TS 29.060 for GTP, and in TS
29.274
                -- for eGTP.
                -- 24.008
```

```
                --
UserCSGInformation          ::= SEQUENCE
{
                cSGId                                           [0] CSGId,
                cSGAccessMode                                   [1] CSGAccessMode,
                cSGMembershipIndication                         [2] NULL OPTIONAL
}
TBCDSTRING ::= OCTET STRING
ISDN-AddressString ::= OCTET STRING
IMEI ::= TBCDSTRING (SIZE(8))
IMSI ::= TBCDSTRING (SIZE(3..8))
maxAddressLength    INTEGER ::= 20
AddressString ::= OCTET STRING (SIZE (1..maxAddressLength))
END
```

SGW Charging OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

SGW Charging CDR Statistics

sgw_charging_cdr counter

```
sgw_charging_cdr{action="close_final",app_name="SMF",cause="abnormalRelease",
cluster="Local",data_center="DC",event="AbnormalRelease",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 1
sgw_charging_cdr{action="close_final",app_name="SMF",cause="normalRelease",
cluster="Local",data_center="DC",event="NormalRelease",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 22
sgw_charging_cdr{action="close_final",app_name="SMF",cause="sGWChange",
cluster="Local",data_center="DC",event="SGWChange",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr{action="close_interim",app_name="SMF",cause="maxChangeCond",
cluster="Local",data_center="DC",event="QoSChange",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr{action="close_interim",app_name="SMF",cause="maxChangeCond",
cluster="Local",data_center="DC",event="ServingNodeChange",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 5
sgw_charging_cdr{action="close_interim",app_name="SMF",cause="timeLimit",
cluster="Local",data_center="DC",event="TimeLimit",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 56
sgw_charging_cdr{action="close_interim",app_name="SMF",cause="volumeLimit",
cluster="Local",data_center="DC",event="VolumeLimit",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr{action="open",app_name="SMF",cause="",cluster="Local",
data_center="DC",event="QoSChange",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr{action="open",app_name="SMF",cause="",cluster="Local",
data_center="DC",event="SGWChange",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr{action="open",app_name="SMF",cause="",cluster="Local",
data_center="DC",event="ServingNodeChange",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
```

```
pdn_type="ipv4v6",service_name="sgw-service"} 5
sgw_charging_cdr{action="open",app_name="SMF",cause="",cluster="Local",
data_center="DC",event="StartAccounting",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 26
sgw_charging_cdr{action="open",app_name="SMF",cause="",cluster="Local",
data_center="DC",event="TimeLimit",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 56
sgw_charging_cdr{action="open",app_name="SMF",cause="",cluster="Local",
data_center="DC",event="VolumeLimit",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 3
```

SGW Charging CDR Container Statistics

sgw_charging_cdr_container counter

```
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="qoSChange",
cluster="Local",data_center="DC",event="QoSChange",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 6
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="recordClosure",
cluster="Local",data_center="DC",event="AbnormalRelease",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 1
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="recordClosure",
cluster="Local",data_center="DC",event="NormalRelease",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 22
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="recordClosure",
cluster="Local",data_center="DC",event="SGWChange",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="recordClosure",
cluster="Local",data_center="DC",event="ServingNodeChange",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 5
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="recordClosure",
cluster="Local",data_center="DC",event="TimeLimit",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 56
sgw_charging_cdr_container{action="close",app_name="SMF",change_condition="recordClosure",
cluster="Local",data_center="DC",event="VolumeLimit",gr_instance_id="1",instance_id="0",
pdn_plmn_type="visitor",pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr_container{action="open",app_name="SMF",change_condition="",cluster="Local",
data_center="DC",event="QoSChange",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 6
sgw_charging_cdr_container{action="open",app_name="SMF",change_condition="",cluster="Local",
data_center="DC",event="SGWChange",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 3
sgw_charging_cdr_container{action="open",app_name="SMF",change_condition="",cluster="Local",
data_center="DC",event="ServingNodeChange",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 5
sgw_charging_cdr_container{action="open",app_name="SMF",change_condition="",cluster="Local",
data_center="DC",event="StartAccounting",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 26
sgw_charging_cdr_container{action="open",app_name="SMF",change_condition="",cluster="Local",
data_center="DC",event="TimeLimit",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"} 56
sgw_charging_cdr_container{action="open",app_name="SMF",change_condition="",cluster="Local",
data_center="DC",event="VolumeLimit",gr_instance_id="1",instance_id="0",pdn_plmn_type="visitor",
pdn_type="ipv4v6",service_name="sgw-service"}
```

SGW Sx Report Statistics

sgw_sx_session_report_stats counter

```
sgw_sx_session_report_stats{app_name="SMF",cluster="Local",data_center="DC",
gr_instance_id="1",instance_id="0",service_name="sgw-service",status="success",
sx_session_report_type="USAR"} 55
```

sgw_sx_usage_report_stats counter

```
sgw_sx_usage_report_stats{app_name="SMF",cluster="Local",data_center="DC",
gr_instance_id="1",instance_id="0",service_name="sgw-service",status="success"}
95
```

GTPP-EP Statistics

gtpp_received_cdrs_total counter

```
gtpp_received_cdrs_total{app_name="SMF",cluster="Local",data_center="DC",dictionary="custom24",
gtpp_profile="pf1",instance_id="0",service_name="gtpp-ep"} 7
```

gtpp_processed_cdrs_total counter

```
gtpp_processed_cdrs_total{app_name="SMF",cluster="Local",data_center="DC",dictionary="custom24",
gtpp_profile="pf1",instance_id="0",service_name="gtpp-ep",status="success"} 7
```

gtpp_batched_cdrs_total gauge

```
gtpp_batched_cdrs_total{app_name="SMF",cluster="Local",data_center="DC",dictionary="custom24",
gtpp_profile="pf1",instance_id="0",service_name="gtpp-ep",status="batch_success"}
 2
```

gtpp_batch_flush_millis_total counter

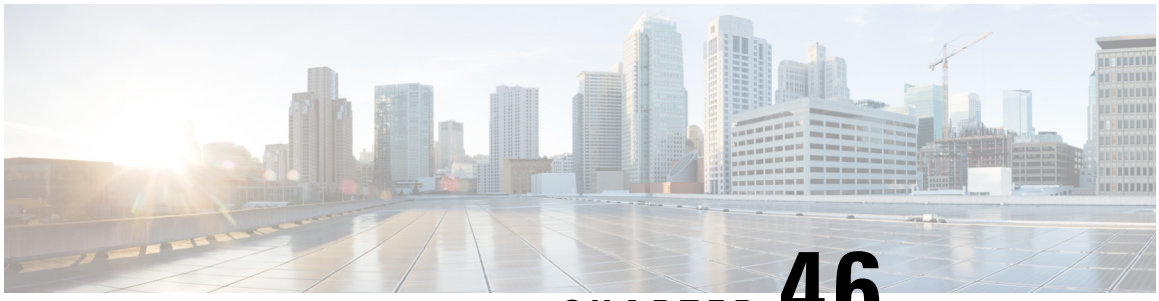
```
gtpp_batch_flush_millis_total{app_name="SMF",cluster="Local",data_center="DC",dictionary="custom24",
gtpp_profile="pf1",instance_id="0",service_name="gtpp-ep",status="batch_success"}
 1126.000588626
```

gtpp_batch_flush_duration_histogram_total counter

```
gtpp_batch_flush_duration_histogram_total{app_name="SMF",bin=">5000ms",cluster="Local",data_center="DC",
dictionary="custom24",gtpp_profile="pf1",instance_id="0",service_name="gtpp-ep",status="batch_success"}
 6
```

gtpp_asn1field_encoding_failures_total

```
gtpp_asn1field_encoding_failures_total{app_name="SMF",cluster="Local",data_center="DC",gtpp_profile="pf1",
dictionary="custom24",asn1_field="ServedIMSI",reason="Constraint
Violation",gr_instance_id="1",service_name="gtpp-ep"}1
```



CHAPTER 46

SGW Relocation Support

- Feature Summary and Revision History, on page 659
- Feature Description, on page 659
- How it Works, on page 660
- SGW Relocation OAM Support, on page 676

Feature Summary and Revision History

Summary Data

Table 230: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 231: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

This feature supports following procedures:

- S1 based SGW Relocation

- X2 based SGW Relocation
- TAU SGW Relocation
- 5G to 4G SGW Relocation

This feature also supports ePCO Indication flag at the PDN level, if it receives this indication in CS Request during Initial attach or PDN connection or SGW relocation.

SGW triggers a Modify Bearer Request to PGW in the following scenario:

- The source MME supports ePCO and the target MME does not support it.
- The target MME supports ePCO and the source MME does not support it.

Note When 4G SGW relocation Create Session Request message receives 5GS Interworking Indication (5GSIWKI), then set SGW relocation type as 5G.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for SGW relocation feature.

X2 Handover SGW Relocation to cnSGW-C Call Flow

This section describes the X2 handover SGW relocation to cnSGW-C call flow.

Figure 121: X2 Handover SGW Relocation to cnSGW-C Call Flow

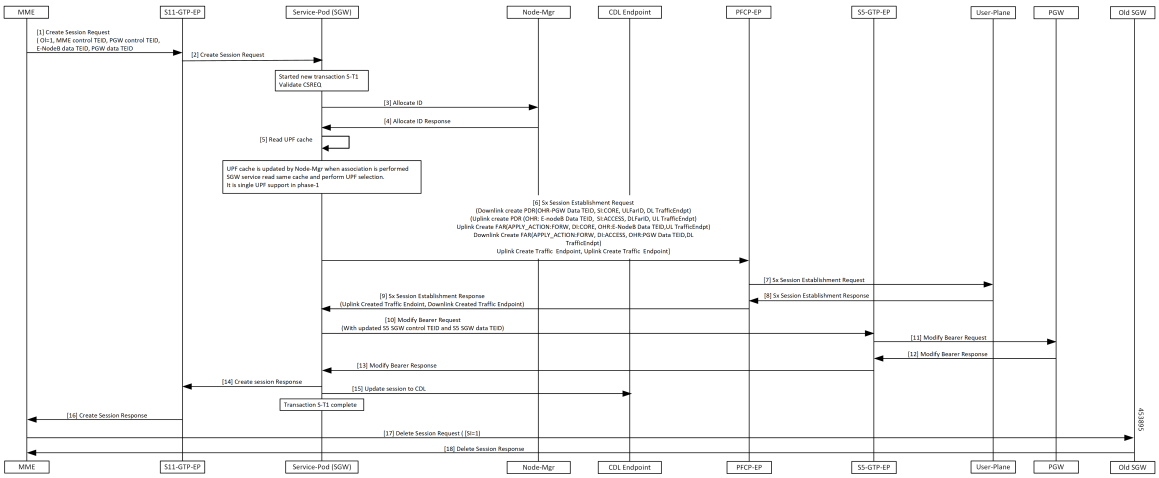


Table 232: X2 Handover SGW Relocation to cnSGW-C Call Flow Description

Step	Description
1	MME sends Create Session Request message to GTPC-EP ingress with the following: • OI flag set • MME Control TEID • PGW Control TEID • eNodeB Data TEID • PGW Data TEID Establishes new transaction at GTPC-EP ingress.
2	SGW service POD receives Create Session Request.
3	SGW service POD Create a new transaction S-T1.
4	Validate Create Session Req.
5	NodeMgr allocates TEID. SGW service POD reads the UPF cache and performs UPF selection.
6	PFCP-EP receives Sx Session Establishment Request from SGW service POD with the uplink and downlink Create PDRs/FARs (Apply Action as Forward)/CTEs.
7	PFCP-EP forwards Sx Session Establishment Request to UPF.
8	PFCP-EP receives Sx Session Establishment Response from UPF with Created CTEs.
9	SGW service POD receives Sx Session Establishment Response from PFCP-EP.
10	Modify Bearer Request with updated S5 SGW Control TEID and S5 SGW Data TEID sent from the SGW service POD to GTPC-EP.
11	PGW receives Modify Bearer Request message from GTPC-EP.
12	GTCP-EP receives Modify Bearer Response from PGW.
13	SGW service POD receives Modify Bearer Response from GTPC-EP.
14	SGW service POD forwards Create Session Response to GTPC-EP ingress.
15	Updated session at CDL. Transaction S-T1 completed.
16	GTPC-EP ingress forwards Create Session Response to MME.
17	MME sends Delete Session Request with SI=1 to old SGW and receives Delete Session Response. Call cleared in old SGW.

S1 Handover SGW Relocation to cnSGW-C Call Flow

This section describes the S1 handover SGW Relocation to cnSGW-C call flow.

Figure 122: S1 Handover SGW Relocation to cnSGW-C Call Flow

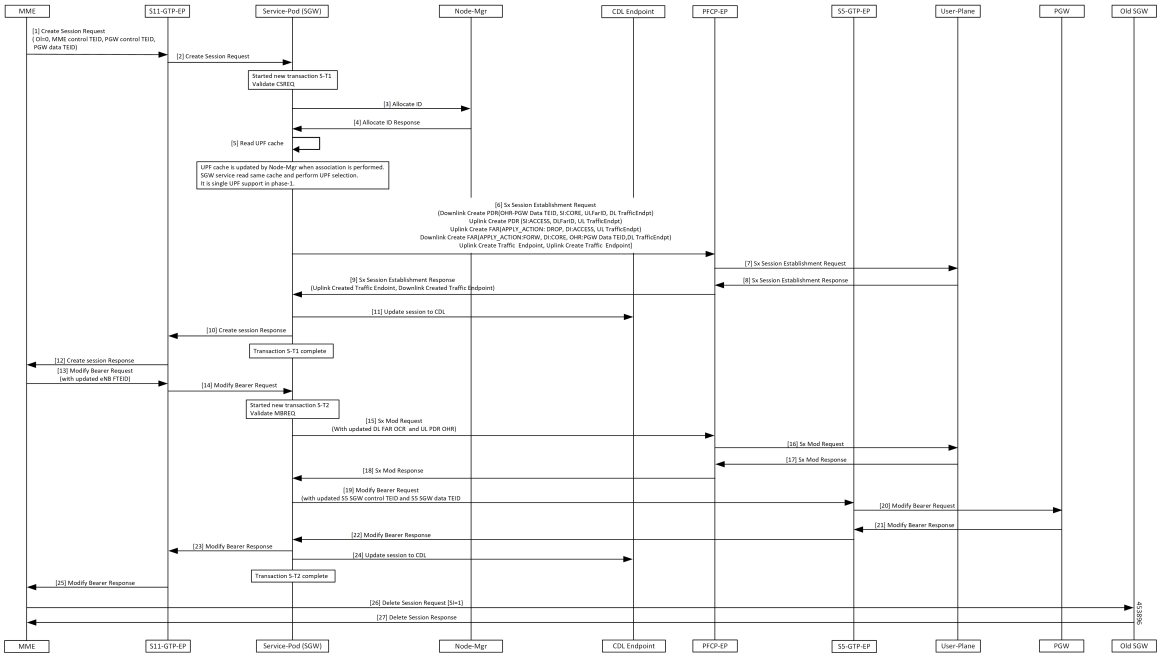


Table 233: S1 Handover SGW Relocation to cnSGW-C Call Flow Description

Step	Description
1	MME sends Create Session Request message to GTPC-EP ingress with the following: • OI flag unset • MME Control TEID • PGW Control TEID • PGW Data TEID Establishes new transaction at GTPC-EP ingress.
2	SGW service POD receives Create Session Request.
3	Create a new transaction S-T1.
4	Validate Create Session Req.
5	NodeMgr allocates TEID. SGW service reads the UPF cache and performs UPF selection.
6	PFCP-EP receives Sx Session Establishment Request from SGW service POD with the following: • Uplink and downlink Create PDRs/FARs (Apply Action as Forward)/CTEs

Step	Description
7	PFCP-EP forwards Sx Session Establishment Request to UPF
8	PFCP-EP receives Sx Session Establishment Response from UPF with Created CTEs.
9	SGW service POD receives Sx Session Establishment Response from PFCP-EP.
10	SGW service POD forwards Create Session Response to GTPC-EP ingress.
11	Updated session at CDL. Transaction S-T1 completed.
12	GTPC-EP ingress forwards Create Session Response to MME.
13	GTPC-EP ingress receives Modify Bearer Req with updated eNodB FTEID from MME.
14	GTPC-EP ingress forwards Modify Bearer Req to SGW service POD. Creates new transaction-T2.
15	PFCP-EP receives Sx Mod Req from SGW service POD with the updated downlink FAR and uplink PDR.
16	PFCP-EP forwards Sx Mod Req to UPF.
17	PFCP-EP receives Sx Mod Response from UPF.
18	SGW receives Sx Mod Response from PFCP-EP.
19	SGW service POD sends Modify Bearer Request with updated S5 SGW Control TEID and S5 SGW Data TEID to GTPC-EP.
20	GTPC-EP forwards Modify Bearer Request to PGW.
21	GTPC-EP receives Modify Bearer Response from PGW.
22	SGW service POD receives Modify Bearer Response from GTPC-EP.
23	GTPC-EP ingress receives Modify Bearer Response from SGW service POD.
24	Session updated at CDL. Transaction S-T2 completed.
25	GTPC-EP ingress forwards Modify Bearer Response to MME.
26	MME sends Delete Session Request with SI=1 to old SGW and receives Delete Session Response. Call cleared in old SGW.

TAU X2 Handover SGW Relocation to cnSGW-C Call Flow

This section describes the TAU X2 handover SGW telocation to cnSGW-C call flow.

Figure 123: TAU X2 Handover SGW Relocation to cnSGW-C Call Flow

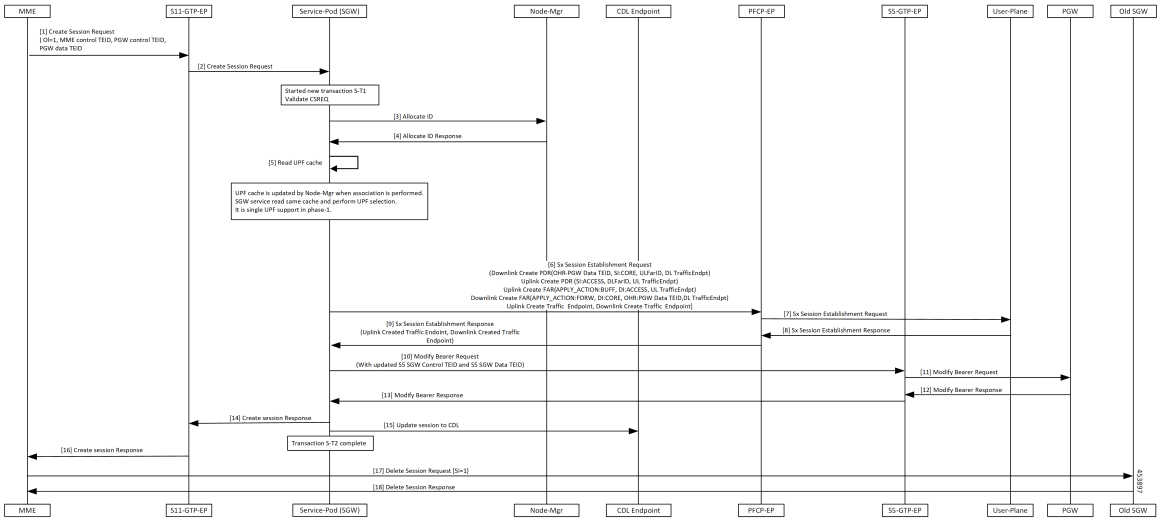


Table 234: TAU X2 Handover SGW Relocation to cnSGW-C Call Flow Description

Step	Description
1	MME sends Create session Req to GTPC-EP ingress with the following: • OI flag set • MME Control TEID • PGW Control TEID • PGW Data TEID Establishes new transaction at GTPC-EP ingress.
2	GTPC-EP ingress forwards Create Session req to SGW service POD.
3	SGW service POD receives Create Session Req. Create a new transaction S-T1.
4	Validate CSReq. NodeMgr performs TEID allocation.
5	SGW service reads UPF Cache and performs UPF selection.
6	PFCP-EP receives Sx Session Establishment Req from SGW service POD with the following: • Uplink and downlink Create PDRs/FARs(ApplyAction as Forward for the uplink FAR)/CTEs.
7	PFCP-EP forwards Sx Session Establishment Req to UPF.
8	PFCP-EP receives Sx Session Establishment Response from UPF.
9	SGW service POD receives Sx Session Establishment Response from PFCP-EP.

Step	Description
10	SGW service POD sends Modify Bearer Req with updated S5 SGW Control TEID and S5 SGW Data TEID to GTPC-EP.
11	GTPC-EP forwards Modify Bearer Request to PGW.
12	GTPC-EP receives Modify Bearer Response from PGW.
13	GTPC-EP forwards Modify Bearer Response to SGW service POD.
14	SGW service POD forwards Create Session Response to GTPC-EP ingress.
15	Session updated at CDL. Transaction S-T1 completed.
16	GTPC-EP ingress forwards Create Session Response to MME.
17	MME sends Delete Session Request with SI=1 sent to old SGW and receives Delete Session Response. Call cleared in old SGW.

X2 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow

This section describes the X2 handover SGW relocation to CN-SGW (Multi PDN) to cnSGW-C call flow.

Figure 124: X2 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow

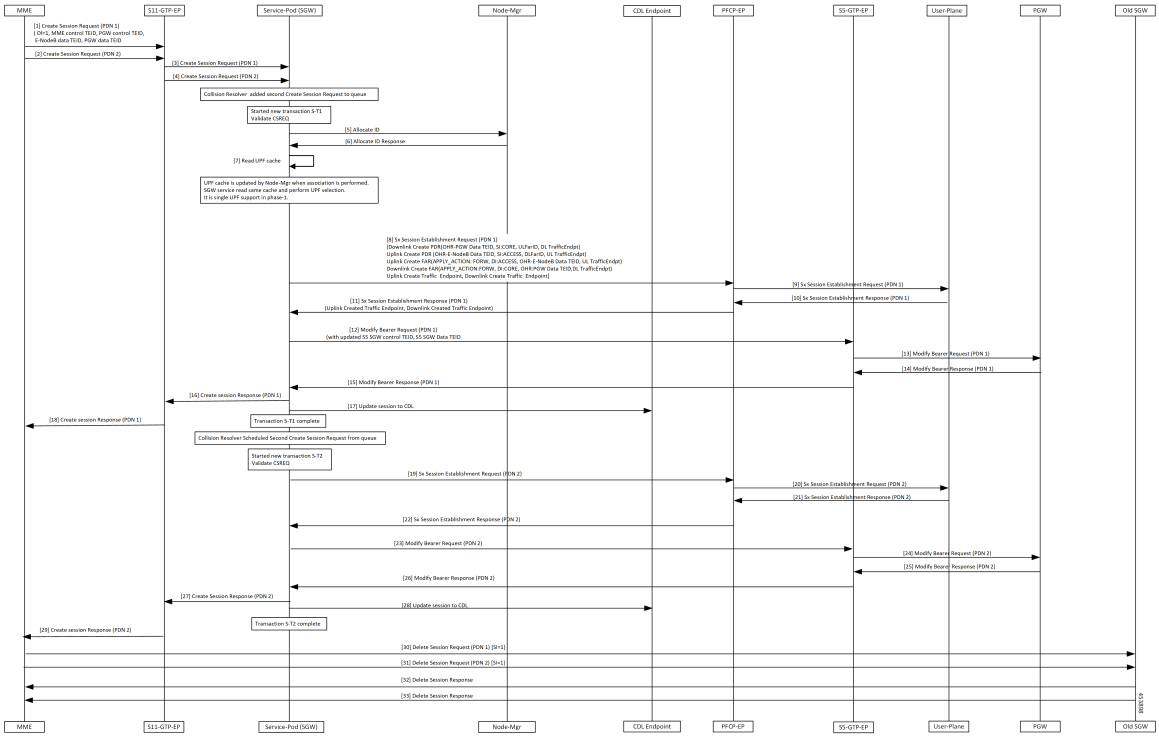


Table 235: X2 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow Description

Step	Description
1, 2	MME sends Create Session Req for both PDNs to GTPC-EP ingress with the following: • OI flag set • MME Control TEID • PGW Control TEID • eNodeB Data TEID • PGW Data TEID Establishes new transactions at GTPC-EP ingress.
3, 4	SGW service POD receives Create Session Req for both PDNs from PFCP-EP ingress. Collision resolver added Create Session Req for PDN 2 in queue.
5	Create a new transaction S-T1 for Create Session Req for PDN1 Validate CSReq.
6	NodeMgr allocates TEID.
7	SGW service POD reads UPF Cache and performs UPF selection.
8	PFCP-EP receives Sx Session Establishment Req from SGW service POD for PDN 1 with the following: • Uplink and downlink Create PDRs/FARs(ApplyAction as Forward)/CTEs.
9	PFCP-EP forwards Sx Session Establishment Req for PDN 1 to UPF.
10	UPF sends Sx Session Establishment Response for PDN 1 to PFCP-EP with Created CTEs.
11	PFCP-EP forwards Sx Session Establishment Response for PDN 1 to SGW service POD.
12	SGW service POD sends Modify Bearer Req for PDN 1 to GTPC-EP with the following: • Updated S5 SGW Control TEID • S5 SGW Data TEID
13	GTPC-EP forwards Modify Bearer Req for PDN 1 to PGW.
14	PGW sends Modify Bearer Response for PDN 1 to GTPC-EP.
15	GTPC-EP forwards Modify Bearer Response for PDN 1 to SGW service POD.
16	SGW service forwards Create Session Response for PDN 1 to GTPC-EP ingress.
17	Session updated at CDL. Transaction S-T1 completed. Collision resolver schedules Create Session Req for PDN 2 from queue.

Step	Description
18	GTPC-EP ingress forwards Create Session Response for PDN 1 to MME.
19 - 27	Repeat steps from 8 to 16 for PDN2.
28	Session updated at CDL. Transaction S-T2 completed.
29	GTPC-EP ingress forwards Create Session Response for PDN 2 to MME.
30 - 33	MME sends Delete Session Request for both PDNs with SI=1 to old SGW and receives Delete Session Response. Call cleared in old SGW.

S1 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow

This section describes the S1 handover SGW relocation to CN-SGW (Multi PDN) to CN-SGW call flow.

Figure 125: S1 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow

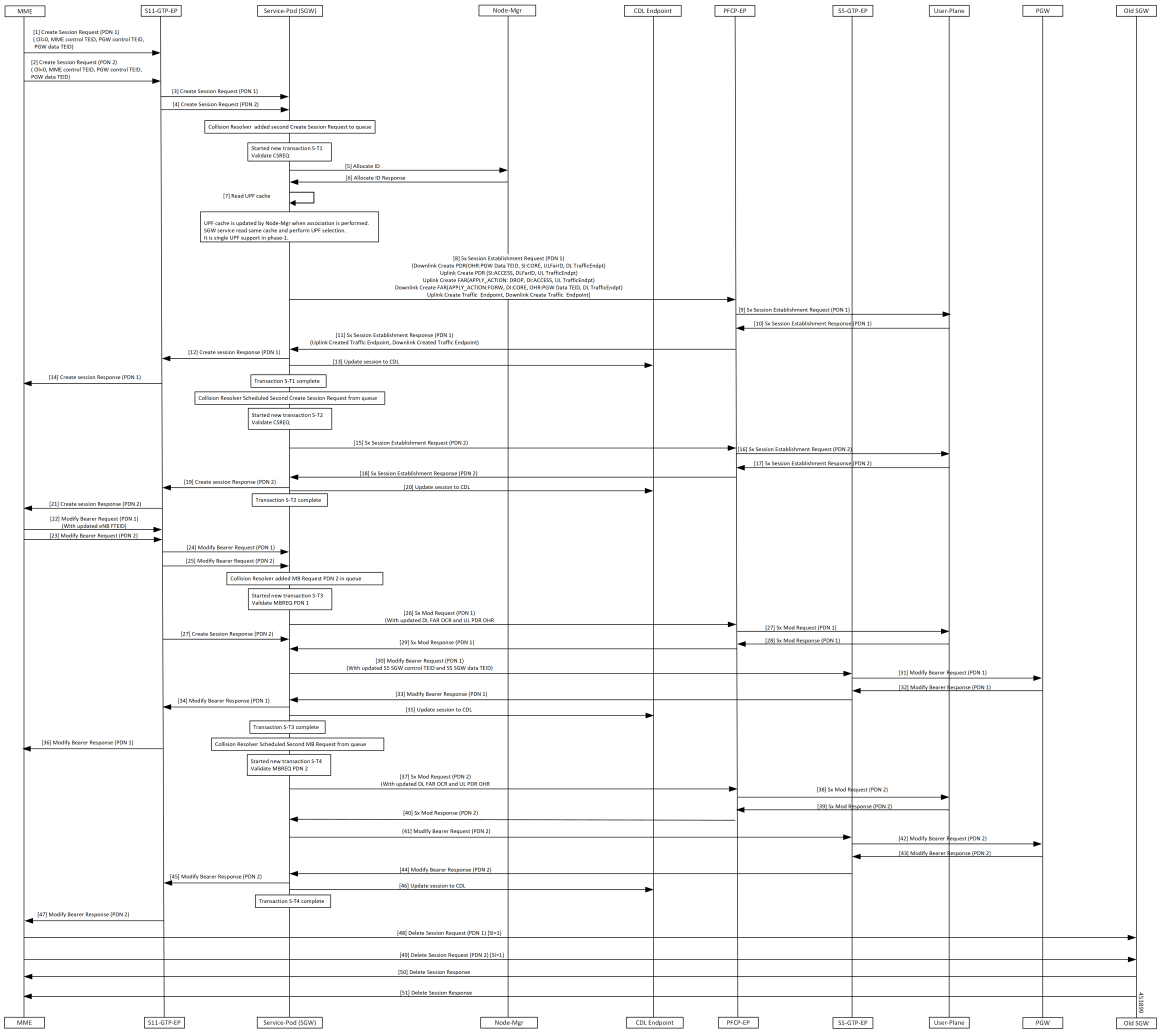


Table 236: S1 Handover SGW Relocation to CN-SGW (Multi PDN) Call Flow Description

Step	Description
1, 2	GTCP-EP ingress receives Create Session Req for both the PDNs with the following: • Ol flag unset • MME control TEID • PGW control TEID • PGW Data TEID Establishes new transaction at GTPC-EP ingress.
3, 4	SGW service POD receives Create Session Req for both the PDNs from GTPC-EP ingress. Collision resolver added Create Session Req for PDN 2 in queue. Create a new transaction S-T1.
5	Validate CSReq.
6	NodeMgr allocates TEID.
7	SGW service reads UPF Cache and performs UPF selection.
8	PFCP-EP receives Sx Session Establishment Req from SGW service POD for PDN 1 with the following: • Uplink and downlink Create PDRs/FARs/CTEs.
9	PFCP-EP forwards Sx Session Establishment Req for PDN 1 to UPF.
10	PFCP-PE receives Sx Session Establishment Response for PDN 1 from UPF with Created CTEs.
11	SGW service POD receives Sx Session Establishment Response for PDN 1 from PFCP-EP.
12	SGW service POD forwards Create Session Response for PDN 1 to GTPC-EP ingress.
13	Session updated at CDL. Transaction S-T1 completed. Collision resolver scheduled Create Session Req for PDN 2 from queue.
14	GTPC-EP ingress forwards Create Session Response for PDN 1 to MME.
15–21	Repeat steps 11–14 for PDN2(S-T2).
22, 23	GTPC-EP ingress receives Modify Bearer Req for both the PDNs with updated eNodB FTEID from MME.
24, 25	GTPC-EP ingress forwards Modify Bearer Req to both PDNs to SGW service POD. Collision resolver added Modify Bearer Req for PDN 2 in the queue. Create a new transaction S-T3.

Step	Description
26	PFCP-EP receives Sx Session Modification Req for PDN 1 from SGW service POD with the updated downlink FAR and uplink PDR.
27	PFCP-EP forwards Sx Session Modification Req for PDN 1 to UPF.
28	UPF sends Sx Session Modification Response for PDN 1 to PFCP-EP.
29	SGW service POD receives Sx Modify Response for PDN 1 from PFCP-EP.
30	GTPC-EP receives Modify Bearer Req for PDN 1 from SGW service POD with the following: • Updated S5 SGW Control TEID and S5 SGW Data TEID.
31	GTPC-EP forwards Modify Bearer Req for PDN 1 to PGW.
32	GTPC-EP receives Modify Bearer Response for PDN 1 from PGW.
33	GTPC-EP forwards Modify Bearer Response for PDN 1 to SGW service POD.
34	SGW service POD forwards Modify Bearer Response for PDN 1 to GTPC-EP ingress.
35	Session updated at CDL.
36	GTPC-EP ingress forwards Modify Bearer Response for PDN 1 to MME.
37	SGW service POD sends Modify Bearer Req for PDN 2 to PFCP-EP. Transaction S-T3 completed. Collision resolver schedules Modify Bearer Req to PDN 2.
38–40	Repeat steps 27,28, 29 for PDN 2.
41	SGW service POD sends Modify Bearer request for PDN 2 to GTPC-EP.
42–47	Repeat steps 31–36 for PDN 2.
48, 49	MME sends Delete Session Request for both the PDNs with SI=1 sent to old SGW.
50	MME receives Delete Session Response. Call cleared in old SGW.

X2 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow

This section describes the X2 handover SGW relocation with bearer context marked for removal call flow.

Figure 126: X2 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow

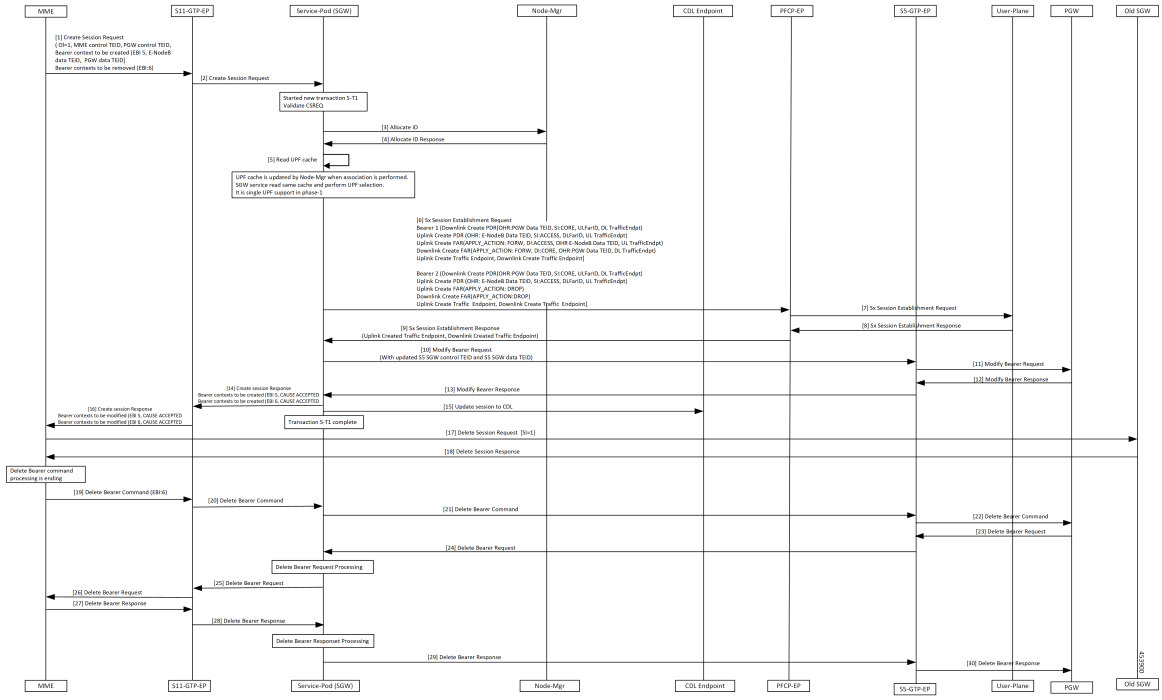


Table 237: X2 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow Description

Step	Description
1	GTPC-EP ingress receives Create Session Req with the following: • OI flag set • MME Control TEID • PGW Control TEID • new Bearer Contexts to create (EBI:5, eNodeB Data TEID and PGW Data TEID) • Bearer context to delete (EBI: 6) Establishes new transaction at GTPC-EP.
2	GTPC-EP ingress forwards Create Session Req to SGW service POD.
3	SGW service POD receives Create Session Req. Create a new transaction S-T1.
4	Validate CSReq. NodeMgr allocates TEID.
5	SGW service POD reads UPF Cache and performs UPF selection.

Step	Description
6	PFCP-EP receives Sx Session Establishment Req from SGW service POD with the following: • Uplink and downlink Create PDRs/FARs (ApplyAction as Forward for EBI 5 and as Drop EBI 6)/CTEs.
7	PFCP-EP forwards Sx Session Establishment Req to UPF.
8	PFCP-EP receives Sx Session Establishment Response from UPF with Created CTEs.
9	SGW service POD receives Sx Session Establishment Response from PFCP-EP.
10	SGW service POD sends Modify Bearer Req with updated S5 SGW Control TEID and S5 SGW Data TEID to GTPC-EP.
11	GTPC-EP forwards Modify Bearer Request to PGW.
12	GTPC-EP receives Modify Bearer Response from PGW.
13	GTPC-EP forwards Modify Bearer Response to SGW service POD.
14	SGW service POD sends Create Session Response to GTPC-EP ingress with cause Accepted for both Bearer Contexts.
15	Session updated at CDL. Transaction S-T1 completed.
16	GTPC-EP ingress forwards Create Session Response to MME.
17	MME sends Delete Session Request with SI=1 sent to old SGW.
18	MME receives Delete Session Response. Call cleared in old SGW.
19	GTPC-EP ingress receives Delete Bearer Command for Bearer Context from MME to delete (EBI 6).
20	SGW service POD receives Delete Bearer Command from GTPC-EP ingress.
21	SGW service POD forwards Delete Bearer Command to GTPC-EP.
22	GTPC-EP forwards Delete Bearer Command to PGW.
23	PGW responds with Delete Bearer Request (EBI 6) to GTPC-EP.
24	GTPC-EP forwards Delete Bearer Request to SGW service POD.
25	SGW service POD processes Delete Bearer Request and sends to GTPC-EP.
26	GTPC-EP ingress forwards Delete Bearer request to MME.
27	MME responds with the Delete Bearer Response to GTPC-EP ingress.
28	GTPC-EP ingress forwards Delete Bearer Response to SGW service POD.
29	SGW service POD processes Delete Bearer Response and sends to GTPC-EP.

Step	Description
30	GTPC-EP forwards Delete Bearer Response to PGW.

S1 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow

This section describes the S1 handover SGW relocation with bearer context marked for removal call flow.

Figure 127: S1 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow

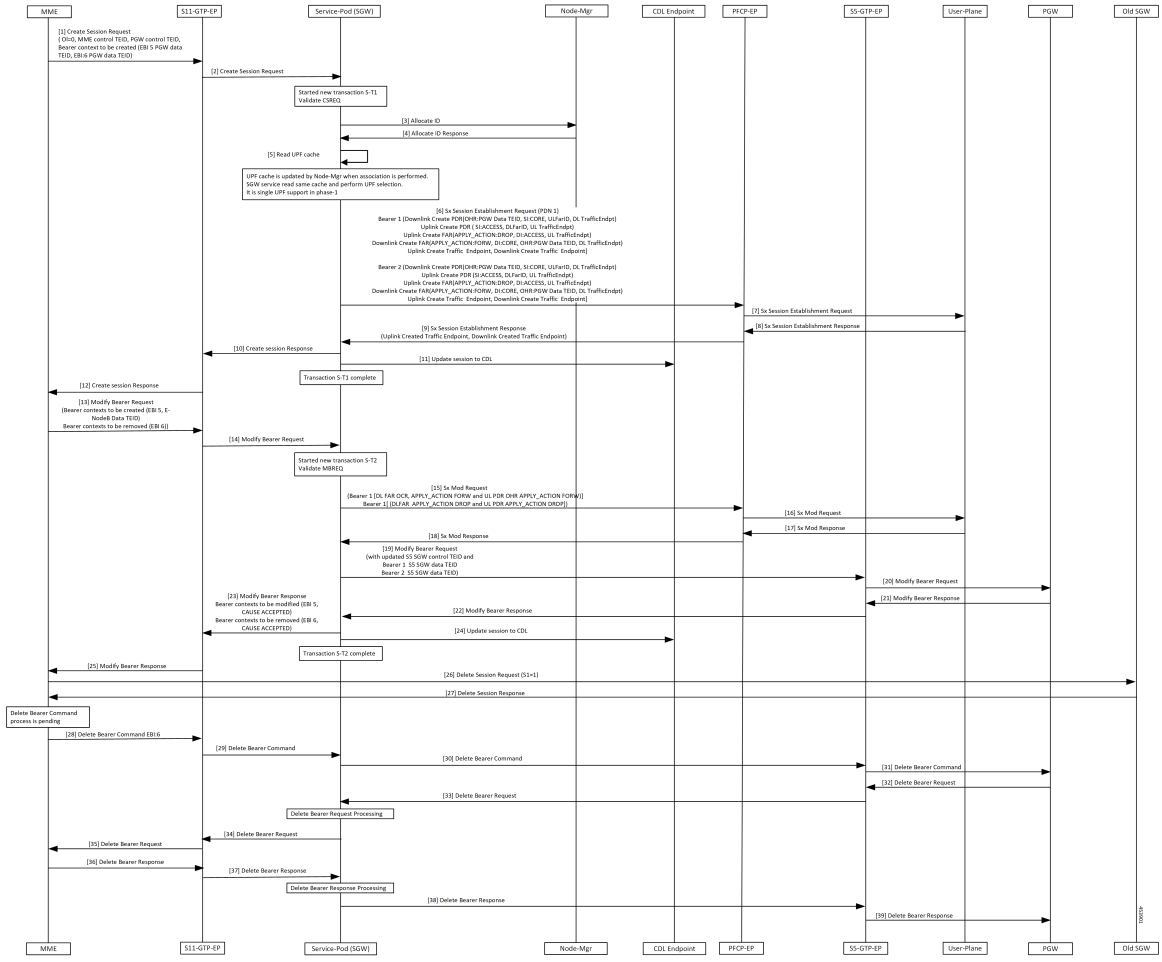


Table 238: S1 Handover SGW Relocation with Bearer Context Marked for Removal Call Flow Description

Step	Description
1	GTPC-EP ingress receives Create Session Request from MME with the following: • OI flag unset • MME Control TEID • PGW Control TEID • Bearer context to create (EBI:5, PGW Data TEID, EBI:6, PGW Data TEID) Establishes new transaction at GTPC-EP ingress.
2	GTPC-EP ingress forwards Create Session req to SGW service POD.
3	SGW service POD receives Create Session Req. Create a new transaction S-T1.
4	Validate CSReq. NodeMgr allocates TEID.
5	SGW service reads UPF cache and performs UPF selection.
6	PFCP-EP receives Sx Session Establishment Req from SGW service with the following: • Uplink and downlink Create PDRs/FARs(ApplyAction as Forward for EBI 5 and as Drop EBI 6)/CTEs.
7	PFCP-EP forwards Sx Session Establishment Req to UPF.
8	PFCP-EP receives Sx Session Establishment Response from UPF with Created CTEs.
9	SGW service receives Sx Session Establishment Response from PFCP-EP.
10	SGW forwards Sx Session Establishment Response to GTPC-EP ingress.
11	Session updated at CDL. Transaction S-T1 completed.
12	GTPC-EP ingress sends Create Session Response to MME.
13	GTPC-EP ingress receives Modify Bearer Req with the following: • Updated eNodeB FTEID with new Bearer Contexts (here EBI 5) and removed (here EBI 6).
14	GTPC-EP forwards Modify Bearer Req to SGW.
15	Create a new transaction S-T2. PFCP-EP receives Sx Session Modification Req from SGW service POD with the following: • Updated downlink FAR and uplink PDR (Apply Action as DROP for Bearer 2).
16	PFCP-EP forwards Sx Session Modification Req forwarded to UPF.

Step	Description
17	UPF sends Sx Modification Response to PFCP-EP
18	PFCP-EP forwards Sx Modification Response to SGW service POD.
19	SGW service POD sends Modify Bearer Req with updated S5 SGW Control TEID and S5 SGW Data TEID to GTPC-EP.
20	GTPC-EP forwards Modify Bearer Request to PGW.
21	GTPC-EP receives Modify Bearer Response from PGW.
22	GTPC-EP forwards Modify Bearer Response to SGW service POD.
23	SGW service POD forwards Create Session Response to GTPC-EP ingress with cause Accepted for both Bearer Contexts.
24	Session updated at CDL. Transaction S-T2 completed.
25	GTPC-EP ingress sends Modify Bearer Response to MME.
26	MME sends Delete Session Request with SI=1 to old SGW.
27	MME receives Delete Session Response. Call cleared in old SGW.
28	GTPC-EP ingress receives Delete Bearer Command from MME for BearerContext to delete (EBI 6).
29	SGW service POD receives Delete Bearer Command from GTPC-EP ingress.
30	SGW service POD forwards Delete Bearer Command to GTPC-EP.
31	GTPC-EP forwards Delete Bearer Command to PGW.
32	PGW responds with Delete Bearer Request (EBI 6) to GTPC-EP.
33	GTPC-EP sends Delete Bearer Request to SGW service POD.
34	SGW service POD processes Delete Bearer Request and sends to GTPC-EP ingress.
35	GTPC-EP ingress sends Delete Bearer request to MME.
36	MME responds with the Delete Bearer Response to GTPC-EP ingress.
37	GTPC-EP ingress receives Delete Bearer response and sends to SGW service POD.
38	SGW service POD processes Delete Bearer Response and forwards to GTPC-EP.
39	GTPC-EP forwards Delete Bearer Response to PGW.

Inter and Intra MME Handover and S1 SGW Relocation with Less Number of Bearer Context Call Flow

This section describes the inter and intra MME handover and S1 SGW relocation with less number of bearer context call flow.

Figure 128: Inter and Intra MME Handover and S1 SGW Relocation with Less Number of Bearer Context Call Flow

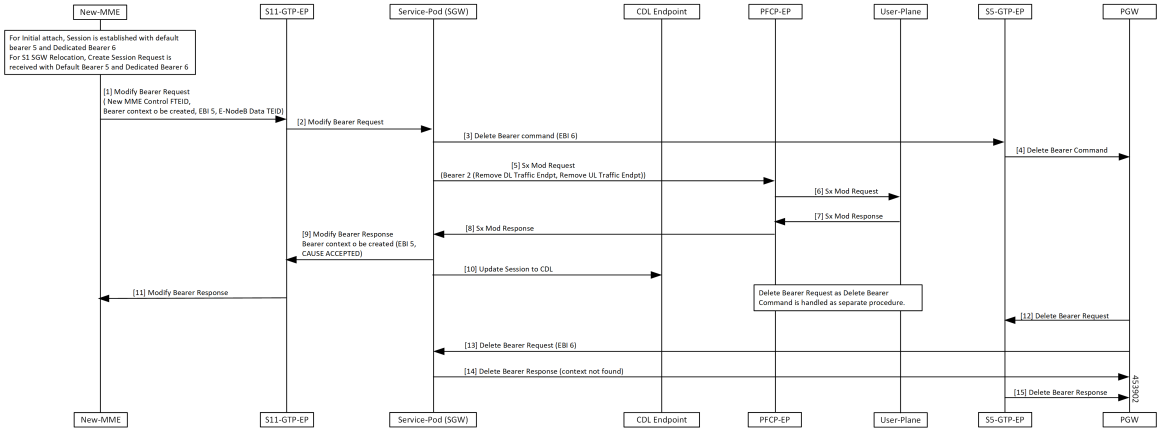


Table 239: Inter and Intra MME Handover and S1 SGW Relocation with Less Number of Bearer Context Call Flow Description

Step	Description
1	Established NEW MME session with the default bearer EBI 5 and dedicated bearer 6. New MME receives Create Session Request for S1 SGW Relocation with Default bearer EBI 5 and dedicated bearer 6. New MME sends Modify Bearer Request to GTPC-EP ingress with the following: • New MME Control TEID • New Bearer Contexts to create • EBI5 • eNodeB Data TEID
2	GTPC-EP ingress forwards Modify Bearer Request to SGW service POD.
3	SGW service POD sends Delete Bearer Command with EBI 6 to GTPC-EP.
4	GTPC-EP forwards Delete Bearer Command to PGW.
5	SGW service POD sends Sx Mod Req to PFCP-EP with Remove DL Traffic Endpoint and Remove UL Traffic Endpoint for Bearer 2.
6	PFCP-EP forwards Sx Mod Req to UPF.
7	UPF sends Sx Mod Response to PFCP-EP.
8	PFCP-EP forwards Sx Mod Response to SGW service POD.

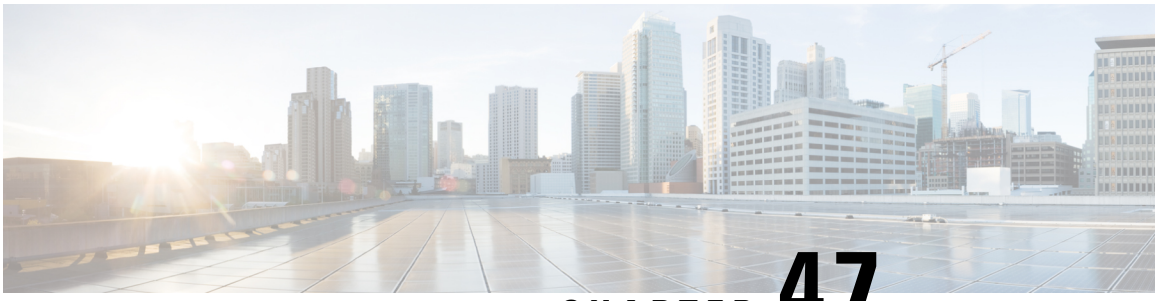
Step	Description
9	SGW service POD sends Modify Bearer Response to GTPC-EP ingress with the following: • Bearer Contexts EBI 5 to modify with cause as *Accepted*
10	Updated session to CDL.
11	GTPC-EP ingress sends Modify Bearer Response to New MME.
12	PGW sends Delete Bearer Request to GTPC-EP.
13	SGW service POD receives Delete Bearer Request with EBI6 from PGW as Delete Bearer Command handled as separate procedure.
14	SGW service POD responds with Delete Bearer Response with cause as *Context Not Found* to GTPC-EP.
15	GTPC-EP forwards Delete Bearer Response to PGW.

Note

- Sx Modify Request message along with Remove DL traffic Endpoint and Remove UL traffic Endpoint is sent as don't confirm message in Legacy CUPS. Sx Modify for MBReq message follows Sx Modify Request and sent to UPF.
- UPF receives the following messages in cnSGW-C.
  - Single Sx Modify Request message for MB Request
  - Remove DL traffic Endpoint
  - Remove UL traffic Endpoint

SGW Relocation OAM Support

This section describes operations, administration, and maintenance information for this feature.



CHAPTER 47

Sx Load/Overload Control Handling

- Feature Summary and Revision History, on page 677
- Feature Description, on page 678
- How it Works, on page 678
- Handling Sx Messages during Peer Overload Scenario, on page 679
- Throttling and Exclusions of Sx Request Messages in Self-Overload State, on page 682
- Sx Load/Overload Control OAM Support, on page 683

Feature Summary and Revision History

Summary Data

Table 240: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 241: Revision History

Revision Details	Release
First introduced.	2021.01.0

Feature Description

This feature handles the load and overload control of messages over Sx interface. There are two scenarios of load and overload control for Sx messages:

- **Peer Overloaded:** When the peer is in overload state, Load control enables the user-plane function to send its load information to the control plane function. This load information is to balance the PFCP session load across the user-plane functions according to their effective loads.

  Overload controls the information for throttling of new session requests towards specific user-plane. UP selection takes place when the user-plane reports LCI (Load control information) and OCI (Overload Control Information).

- **Self-Overload State:** During self-protection state, upon receiving an incoming message, cnSGWc verifies the corresponding DNN, ARP, QCI, Message Priority values. Based on this, cnSGWc throttles the messages or excludes them from throttling as per the overload exclude profile configuration.

How it Works

The load and overload control over Sx interface consists of two parts:

- **Peer Overload Handling:** Handling the messages when UPF is overloaded.
- **Self Overload Handling:** Handling the messages when cnSGWc is overloaded.

Node Feature Support

As per 3GPP standard:

- CP informs load and overload feature to the user-plane.
- User-plane decides to send load or overload information towards the CP peer or not.

Configure load and overload feature at CP as a part of PFCP Sxa endpoint node feature. This configuration in turn communicates to UP during Sx Association Response message or Sx Association Update Request message when change in configuration occurs.

The CP Function Feature IE indicates the supported CP function features. This IE contains features which have (system-wide) UP function behavior impact.

Note If CP does not support load or overload feature through CLI then it ignores the user-plane reported load or overload information for the UP selection process.

UP Selection

UP selection occurs as per LCI value only whereas throttling occurs as per OCI value only (Specified in 3GPP standards).

Per Peer Level LCI and OCI display:

```
show peers | tab | exclude rest
                                                
```

Failure-handling profile is associated with UPF-Group.

Reselection of UPF follows the UPF selection process and considers the retries count to different UPF from profile configuration.



Note Currently, only `PFCP_ENTITY_IN_CONGESTION(74)` is supported as cause code for retry and reselection of user-plane as part of this feature.

Configuring the Sx Load/Overload Feature

This section describes how to configure Sx Load/Overload.

Use the following commands to configure Sx Load/Overload configuration.

```
config
 instance instance-id instance_id
  endpoint endpoint_name
   interface interface_name
    supported-features [ load-control | overload-control ]
   exit
  exit
```

NOTES:

- **endpoint** *endpoint_name* - Specify the endpoint name.
- **interface** *interface_name* - Specify the interface name.
- **supported-features [load-control | overload-control]** - Enable load/overload control.

Sample Configuration

Following is a sample configuration.

```
configure
 instance instance-id 1
  endpoint pfcp
   interface sxa
    supported-features load-control overload-control
   exit
```

Verifying Sx Load/Overload Configuration

Use the following `show` command to view the Sx load/overload configuration.

```
show running-config instance instance-id 1 endpoint
instance instance-id 1
endpoint pfcp
interface sxa
supported-features load-control overload-control
exit
exit
```

Configuring Failure Handling Profile

This section describes how to configure failure handling profile.

Use the following commands to configure failure handling profile.

```
config
 profile failure-handling failure-handling_profile_name
   interface interface_name
    message message_type
      cause-code cause_code
        action action_type
          max-retry max_retry_count
      exit
    exit
   exit
 profile upf-group upf-group_profile_name
  failure-profile profile_name
 exit
```

NOTES:

- **profile failure-handling** *failure-handling_profile_name* - Specify the failure-handling profile name.
- **interface** *interface_name* - Specify the interface name.
- **message** *message_type* - Specify the message type.
- **cause-code** *cause_code* - Specify the cause ID (range of 2-255) or range of cause IDs (range of 2-255) separated by either '-' or ',' or both.

  -Or-

  Must be one of the following:

  - no-resource-available
  - no-response-received
  - pfcp-entity-in-congestion
  - reject
  - service-not-supported
  - system-failure

- **action** *action_type* - Specify the action type for the cause. Must be one of the following:
  - retry-terminate
  - terminate
- **max-retry** *max_retry_count* - Specify the maximum retry count for the retry-terminate action. Must be an integer in the range of 0-5. Default value is 1.
- **profile upf-group** *upf-group_profile_name* - Specify the UPF group profile name.
- **failure-profile** *profile_name* - Specify the UPF failure profile name.

Sample Configuration

Following is the sample configuration:

```
profile failure-handling fh1
 interface sxa
  message SessionEstablishmentReq
   cause-code pfcp-entity-in-congestion action terminate
  exit
 exit
exit
profile failure-handling fh2
 interface sxa
  message SessionEstablishmentReq
   cause-code 74 action retry-terminate max-retry 3
  exit
 exit
exit
profile upf-group g1
 failure-profile fh1
exit
profile upf-group g2
 failure-profile fh2
exit
```

Throttling and Exclusions of Sx Request Messages in Self-Overload State

Table 242: Feature History

Feature Name	Release Information	Description
Handling the Incoming Sx Messages at Self-overload State	2024.04.1	The load and overload control capability of cnSGWc is enhanced to handle the incoming Sx message during the self-overload state. This feature allows the cnSGWc to throttle the incoming DLDR session report request messages in the self-protection mode. **Default Setting:** Disabled- Configuration required to enable.

Throttling and Exclusion of Incoming Sx Request Messages

cnSGWc receives DDLR session report request messages from UPF whenever UPF receives any downlink for an idle session. When the cnSGWc is in the self-protection mode and if the overload exclude profile is configured, cnSGWc verifies the incoming DDLR session report messages for DNN, ARP, QCI, and message priority values.

To configure the overload exclude profile for the Sxa interface, see the Configure the Overload Exclude Profile section.

In case of a match or the session is identified as a WPS or an emergency session, cnSGWc processes the incoming DLDR session report request message, otherwise rejects it with **Cause 74 (PFCP Entity in Congestion)** and initiates an Internal DDN Self-protection Timer.

The Internal DDN Self-protection Timer works in the following way:

- If before Internal DDN Self-protection Timer expires, the cnSGWc moves out of the self-protection state, cnSGW sends the Downlink Data Notification message toward MME.
- If after Internal DDN Self-protection Timer expiry, the cnSGWc remains in the self-protection state, cnSGW sends the Sx Modification Request toward the UPFs with DROBU flag and Downlink FAR apply action as BUFFER.
- If the UE moves out of the idle state (MBReq received) before the DDN Self-protection Timer expires, then cnSGW stops the DDN Self Protection Timer and starts processing MBReq.

Note cnSGWc processes DLDR Session Report Request for emergency, WPS sessions, and Session Report Requests other than DLDR as usual in the self-protection mode.

Throttling and Exclusion of Outgoing Sx Request Messages

In the self-overload state, cnSGWc throttles the outgoing request messages on the Sxa interface from cnSGWc indirectly, as the throttling is applied on incoming GTPC interfaces. cnSGWc processes the messages that are excluded from throttling over the GTPC interface, over the Sxa interface as well.

Sx Load/Overload Control OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

UE Disconnect Statistics

```
sgw_ue_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id=
"0",reason="sx_no_resource_available",service_name="sgw-service"} 1
```

```
sgw_ue_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id=
"0",reason="userplane_requested_termination",service_name="sgw-service"} 1
```

PDN Disconnect Statistics

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id=
"0",pdn_type="ipv4",rat_type="EUTRAN",reason="sx_no_resource_available",service_name="sgw-service"}
 1
```

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id=
"0",pdn_type="ipv4v6",rat_type="EUTRAN",reason="userplane_requested_termination",service_name="sgw-service"}
 1
```

SGW Service Statistics

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="sx_oci_throttling_reject",
instance_id="0",interface="interface_sgw_ingress",reject_cause="no_resources_available",service_name=
"sgw-service",sgw_procedure_type="initial_attach",status="rejected",sub_fail_reason=""}
 1
```

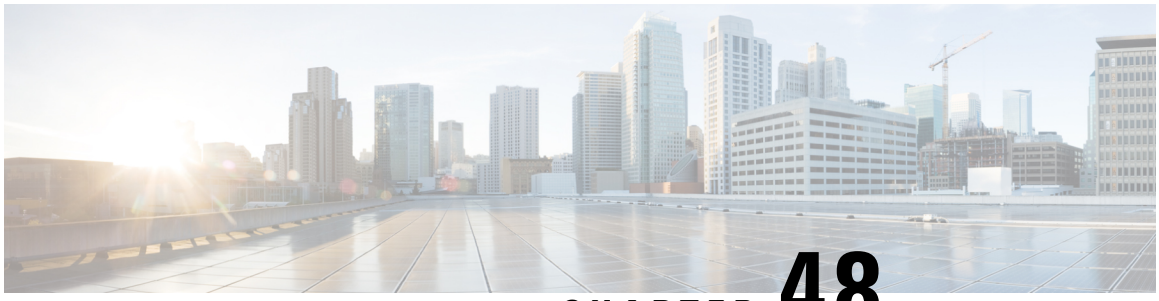
```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="",instance_id="0",
interface="interface_sgw_egress",reject_cause="",service_name="sgwservice",sgw_procedure_type=
"upf_initiated_deletion",status="attempted",sub_fail_reason=""} 1
```

```
sgw_service_stats{fail_reason="sx_cause_fail",interface="interface_sgw_ingress",reject_cause=
"service_denied",sub_fail_reason="pfcp_entity_in_congestion",sgw_procedure_type="initial_attach",
status="rejected"}
```

Session Report Request Throttling Statistics

A new status string **validation_failurePFCP_Entity_In_Congestion** is added in the existing statistics **sgw_sx_session_report_stats**. cnSGWc populates the session report statistics with this status value in case DLDR session report request is rejected in self-protection mode.

```
sgw_sx_session_report_stats{app_name="smf",cluster="Local",data_center="DC",gr_instance_id="1",
instance_id="0",reason="",service_name="sgw-service",status="validation_failurePFCP_Entity_In_Congestion",
sx_session_report_type="DLDR_NO_ARP"} 1
```



CHAPTER 48

Stale Session Handling and Clearing

- Feature Summary and Revision History, on page 685
- Feature Description, on page 686
- How it Works, on page 686
- Feature Configuration, on page 688
- OAM Support, on page 689

Feature Summary and Revision History

Summary Data

Table 243: Summary Data

Applicable Product or Functional Area	cnSGW-C
Applicable Platform	SMI
Feature Default Setting	Disabled - Configuration required to enable
Related Documentation	

Revision History

Table 244: Revision History

Revision Details	Release
First introduced	2021.02.3

Feature Description

The cnSGW-C supports identifying and clearing stale sessions using a *session-stale-timer* parameter in the SGW Profile configuration. An example of a stale session is one that is inactive and not read or modified for a specific period of time.

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flows for this feature.

Timer Expiry Handling Call Flow

This section describes the Timer Expiry Handling call flow.

Figure 129: Timer Expiry Handling Call Flow

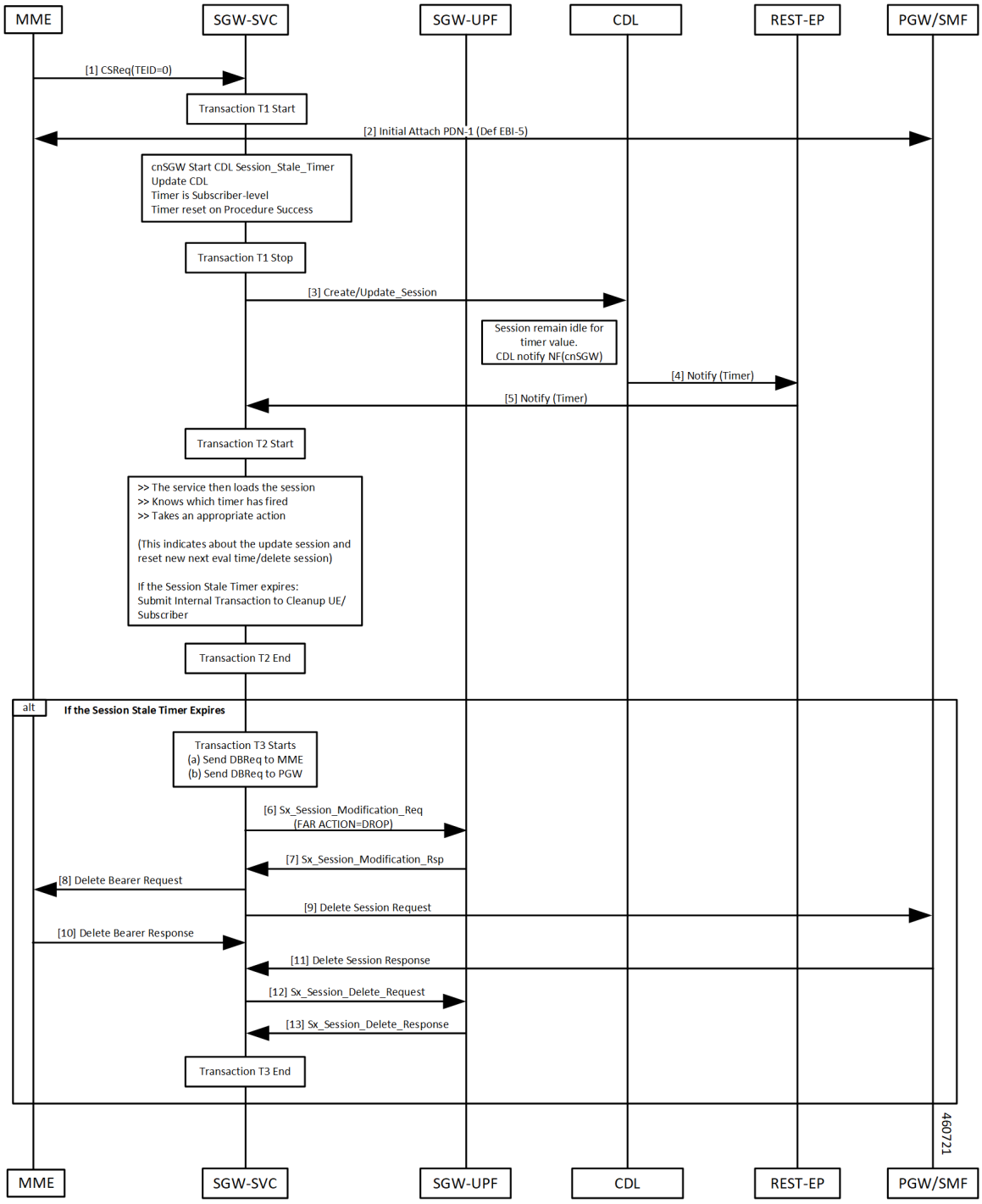


Table 245: Timer Expiry Handling Call Flow Description

Step	Description
1, 2, 3	The initial attach and Session Stale Timer updates are sent to CDL and the timer starts. The sequence is as follows: • Initial Attach Success • Session Stale Timer created • CDL updates done. • CDL starts Time (eval time) and waits for update session. **Note** Session Stale Timer is Reset or Restart, when any of the activity or transaction happens on the Control Plane (cnSGW-C).
4, 5	The Timer expires on CDL pod, and the timer sends notification to cnSGW-C. The sequence is as follows: • If no Session update received for eval timer duration • Timer Expiry on CDL pod • CDL sends Timer Notification to cnSGW-C
6–13	The session cleanup is activated, when the Session Stale Timer expires, and the timer reset isn't required. The sequence is as follows: • Receives Timer Notification on cnSGW-C • If the Timer Notification is for Session Stale Timer and if the timer reset isn't required, it starts UE session clean-up activities. • Sends Sx_Session_Modification_Req to UPF to set FAR Action=Drop, Sx_Session_Modification response received • Sends Delete Bearer Request towards MME • Sends Delete Session Request toward PGW • Sends Sx_Session_Delete_Request toward UPF to clean up User Plane data

Feature Configuration

To configure this feature, use the following configuration:

```
config
  profile sgw sgw_profile_name
  session-stale-timer session_stale_timer
  end
```

NOTES:

- **session-stale-timer** *session_stale_timer* —Specify the maximum number of seconds for which a session can remain idle without any signaling or event, after which the session will be terminated.

  The *session_stale_timer* value must be in the range of 0–4294967295, and must be greater than the **setup-timeout** and **session-idle-timeout** timer values.

  To disable the session-stale-timer configuration, set it to 0.

Configuration Example

The following is an example configuration.

```
config
   profile sgw sgw1
   session-stale-timer 120
   end
```

Configuration Verification

To verify the configuration:

```
show running-config profile sgw sgw1
session-stale-timer 120
```

The output of the show command includes the following field:

session-stale-timer—Indicates the maximum number of seconds for which a session can remain idle without any signaling or event, after which the session is terminated.

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

The following statistics are supported for this feature

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",
fail_reason="",gr_instance_id="1",instance_id="0",interface="interface_sgw_egress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="stale_session_initiated_deletion",status="attempted",
sub_fail_reason=""} 1
```

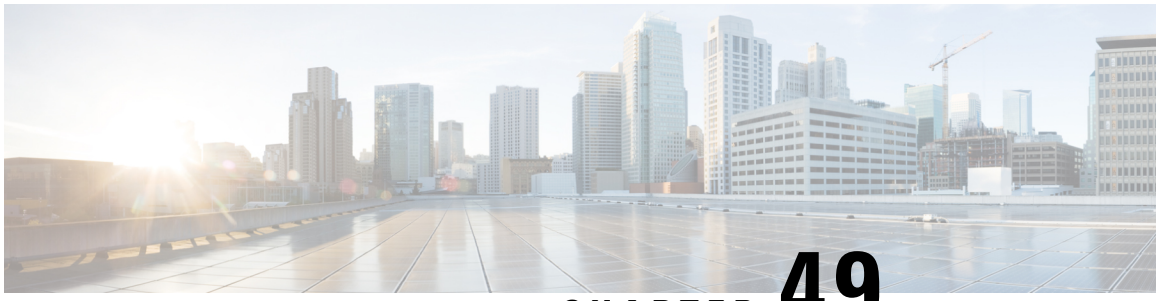
```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_egress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="stale_session_initiated_deletion",status="success",
sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_ingress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="stale_session_initiated_deletion",status="attempted",
sub_fail_reason=""} 1
```

```
sgw_service_stats{app_name="smf",cluster="Local",data_center="DC",fail_reason="",
gr_instance_id="1",instance_id="0",interface="interface_sgw_ingress",reject_cause="",
service_name="sgw-service",sgw_procedure_type="stale_session_initiated_deletion",status="success",
sub_fail_reason=""} 1
```

```
sgw_ue_disconnect_stats{app_name="smf",cluster="Local",data_center="DC",
gr_instance_id="1",instance_id="0",reason="stale_session_init_disconnect",
service_name="sgw-service"} 1
```

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="Local",data_center="DC",
gr_instance_id="1",instance_id="0",pdn_type="ipv4",rat_type="EUTRAN",
reason="stale_session_init_disconnect",service_name="sgw-service"} 1
```



CHAPTER 49

Support for CSFB Procedures Suspend and Resume

- Feature Summary and Revision History, on page 691
- Feature Description, on page 691
- How it Works, on page 692

Feature Summary and Revision History

Summary Data

Table 246: Summary Data

Applicable Products or Functional Area	cnSGW-C
Applicable Platforms	SMI
Feature Default Setting	Enabled – Always-on
Related Documentation	Not Applicable

Revision History

Table 247: Revision History

Revision Details	Release
First introduced.	2022.04.0

Feature Description

Circuit Switched Fall Back (CSFB) enables the UE to camp on an EUTRAN cell and originate or terminate voice calls through a forced switchover to the circuit-switched (CS) domain or other CS-domain services,

such as Location Services (LCS) and supplementary services. Also, SMS delivery through the CS core network is realized without CSFB. As LTE EPC networks are not meant to directly anchor CS connections, when any CS voice services are initiated, any PS-based data activities on the EUTRAN network get suspended (that is, either the data transfer is suspended, or the packet switched connection is handed over to the 2G/3G network). The data activities can further be resumed to enable telephony and SMS services for LTE operators that do not plan to deploy IMS packet switched services at the initial service launch.

cnSGW-C supports the following CSFB Messaging services on an S11 interface over GTPC:

- Suspend Notification
- Suspend Acknowledgment
- Resume Notification
- Resume Acknowledgment

Along with the Resume procedure, it is possible that MME can send a nonempty Modify Bearer Request. In the suspended state, cnSGW-C supports handling the Modify Bearer Request by considering it as an implicit resume procedure.

How it Works

This section describes how this feature works.

cnSGW-C forwards Suspend Notification messages to the PGW to suspend downlink data for non-GBR traffic. Later, when the UE completes the CS services and moves back to E-UTRAN, the MME sends a Resume Notification message to cnSGW-C which forwards the message to the PGW. The downlink data traffic resumes thereafter.

Support for Empty Modify Bearer Request for Resume

In the suspended state, cnSGW-C handles the Modify Bearer Request by considering it as an implicit resume procedure. SGW resumes and forwards the empty Modify Bearer Request to PGW. If eNodeB teid exists in MBR, cnSGW-C sets the Downlink Far Action as Forward. If eNodeB teid does not exist in MBR, cnSGW-C sets the Downlink Far Action as Buffer.

Call Flows

This section describes the key call flow for this feature.

Suspend Notification Call Flow

This section describes the Suspend Notification call flow.

Figure 130: Change Notification Request Call Flow

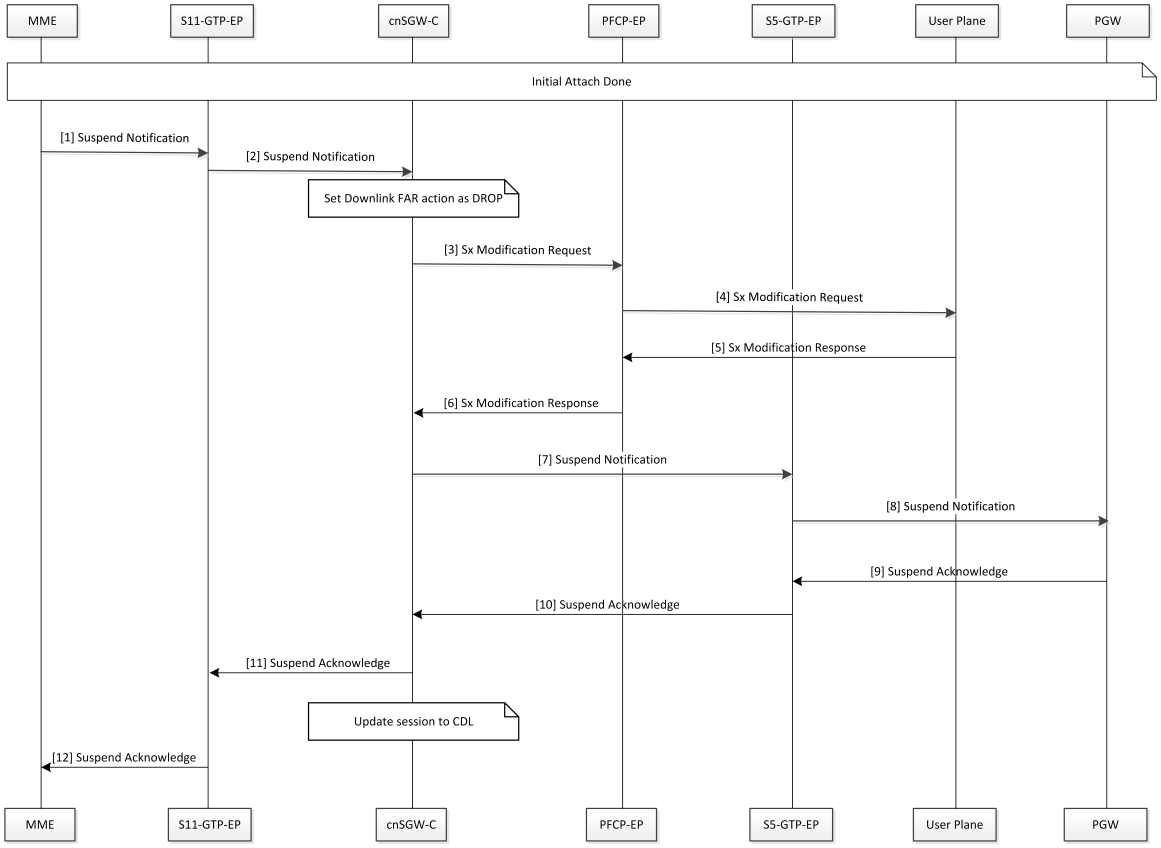


Table 248: Suspend Notification Call Flow Description

Step	Description
1	MME sends the Suspend Notification to the S11-GTP-EP pod.
2	The S11-GTP-EP pod sends the Suspend Notification to the cnSGW-c.
3	cnSGW-c updates the Download FAR action Drop by sending the Sx Session Modification Request to the SGW-U with the FAR action set as DROP. cnSGW-c sends the Sx Modification Request to the PFCP-EP.
4	PFCP-EP sends the Sx Modification Request to the User Plane.
5	The user plane sends the Sx Modification Response to the PFCP-EP.
6	PFCP-EP sends the Sx Modification Response to the cnSGW-C.
7	cnSGW-C sends the Suspend Notification to the S5-GTP-EP pod.
8	The S5-GTP-EP pod sends the Suspend Notification to the PGW.
9	PGW sends the Suspend Acknowledgment to the S5-GTP-EP pod.

Step	Description
10	The S5-GTP-EP pod sends the Suspend Acknowledgment to the cnSGW-C.
11	cnSGW-C sends the Suspend Acknowledgment to the S11-GTP-EP pod. cnSGW-C updates the session to CDL.
12	The S11-GTP-EP pod sends the Suspend Acknowledgment to the MME.

Resume Notification Call Flow

This section describes the resume notification call flow.

Figure 131: Resume Notification Call Flow

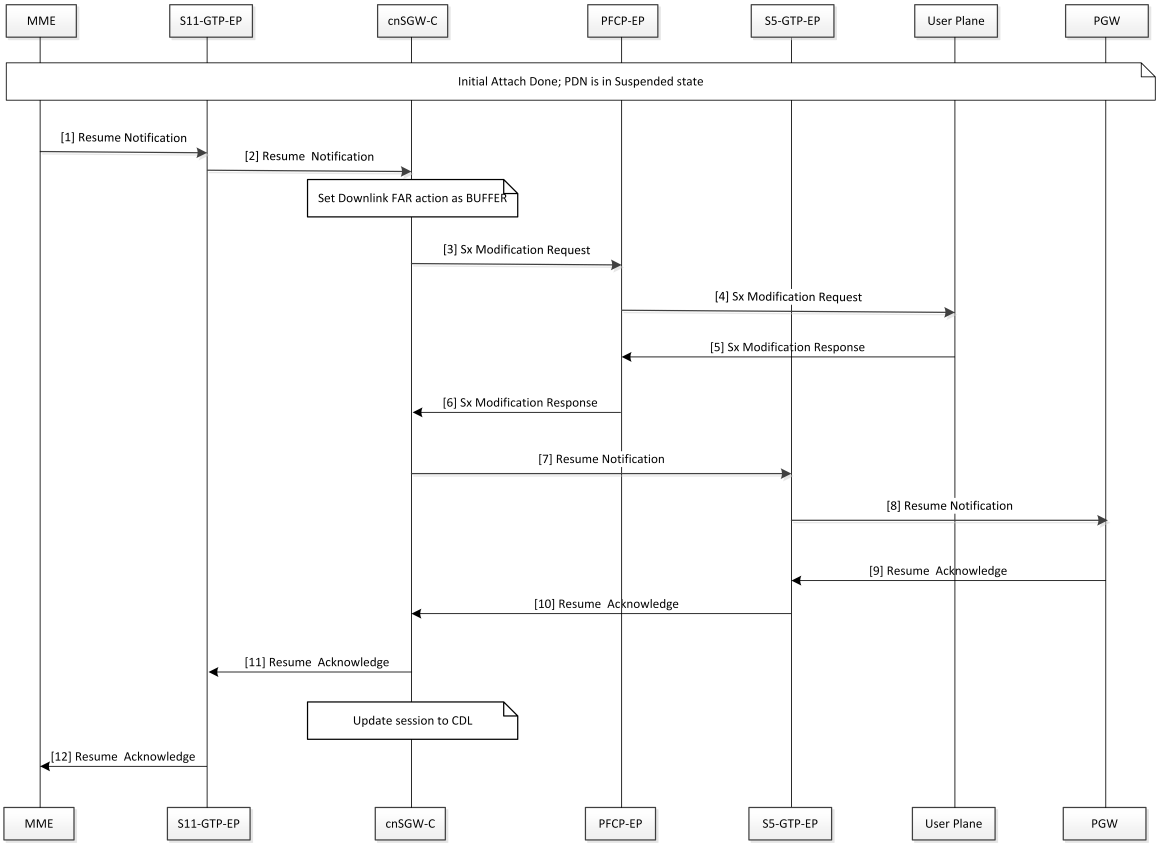


Table 249: Resume Notification Call Flow Description

Step	Description
1	MME sends the Resume Notification to the S11-GTP-EP pod.
2	The S11-GTP-EP pod sends the Resume Notification to cnSGW-c.

Step	Description
3	cnSGW-c updates the Download FAR action Buffer by sending the Sx Session Modification Request to the SGW-U with the FAR action set as BUFFER. cnSGW-c sends the Sx Modification Request to the PFCP-EP.
4	PFCP-EP sends the Sx Modification Request to the User Plane.
5	The User Plane sends the Sx Modification Response to the PFCP-EP.
6	PFCP-EP sends the Sx Modification Response to the cnSGW-C.
7	cnSGW-C sends the Resume Notification to the S5-GTP-EP pod.
8	The S5-GTP-EP pod sends the Resume Notification to the PGW.
9	PGW sends the Resume Acknowledgment to the S5-GTP-EP pod.
10	The S5-GTP-EP pod sends the Resume Acknowledgment to the cnSGW-C.
11	cnSGW-C sends the Resume Acknowledgment to the S11-GTP-EP pod. cnSGW-C updates the session to CDL.
12	The S11-GTP-EP pod sends the Resume Acknowledgment to the MME.

Empty Modify Bearer Request for Resume Call Flow

This section describes the Empty Modify Bearer Request for Resume call flow.

Figure 132: Empty Modify Bearer Request for Resume Call Flow

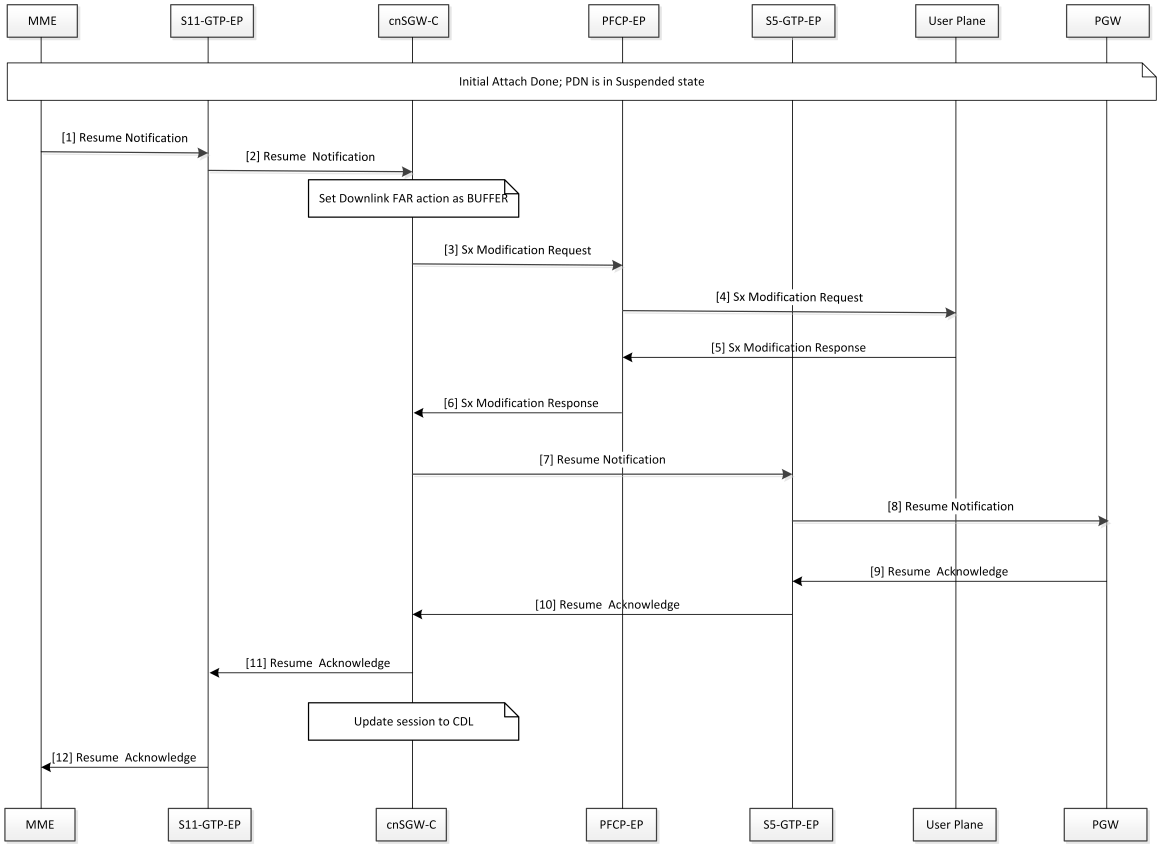
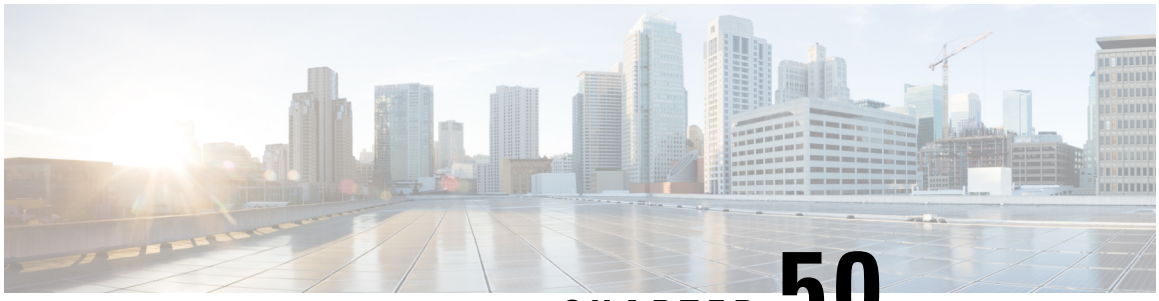


Table 250: Empty Modify Bearer Request for Resume Call Flow Description

Step	Description
1	MME sends the Modify Bearer Request to the S11-GTP-EP pod.
2	The S11-GTP-EP pod sends the Modify Bearer Request to the cnSGW-C. cnSGW-C considers the request as the implicit resume of the suspended bearers in the SGW.
3	If eNodeB exists, cnSGW-C updates Set Downlink Far action as Forward. Else cnSGW-C updates Set Downlink Far action as BUFFER. cnSGW-C sends the Sx Modify Request to the PFCP-EP pod.
4	PFCP-EP sends the Sx Modify Request to the User Plane.
5	The User Plane sends the Sx Modify Response to the PFCP-EP.
6	PFCP-EP sends the Sx Modify Response to the cnSGW-C.
7	cnSGW-C sends the Modify Bearer Request to the S5-GTP-EP pod. The implicit resume for PGW can contain zero or more IEs.

Step	Description
8	The S5-GTP-EP pod sends the Modify Bearer Request to the PGW.
9	PGW sends the Modify Bearer Response to the S5-GTP-EP pod.
10	The S5-GTP-EP pod sends the Modify Bearer Response to the cnSGW-C.
11	cnSGW-C sends the Modify Bearer Response to the S11-GTP-EP pod.
12	The S11-GTP-EP pod sends the Modify Bearer Response to the MME.



CHAPTER 50

Update Bearer Request and Response

- Feature Summary and Revision History, on page 699
- Feature Description, on page 699
- How it Works, on page 700

Feature Summary and Revision History

Summary Data

Table 251: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 252: Revision History

Revision Details	Release
First introduced.	2020.04

Feature Description

In this release, cnSGW-C supports only relay of update bearer request (which can contain TFT change, QCI change or APB – AMBR change) from PGW towards MME. When MME sends response to cnSGW-C, it relays update bearer response towards PGW.

This release doesnt support signaling towards User Plane.

Standards Compliance

The Update Bearer Request and Response Support feature complies with the following standards:

- *3GPP TS 23.401*
- *3GPP TS 23.214*
- *3GPP TS 29.274*
- *3GPP TS 29.244*

How it Works

This section describes how this feature works.

Call Flows

This section describes the key call flow of Update Bearer Request and Response feature.

Figure 133: Bearer Request/Response without UP Signaling Call Flow

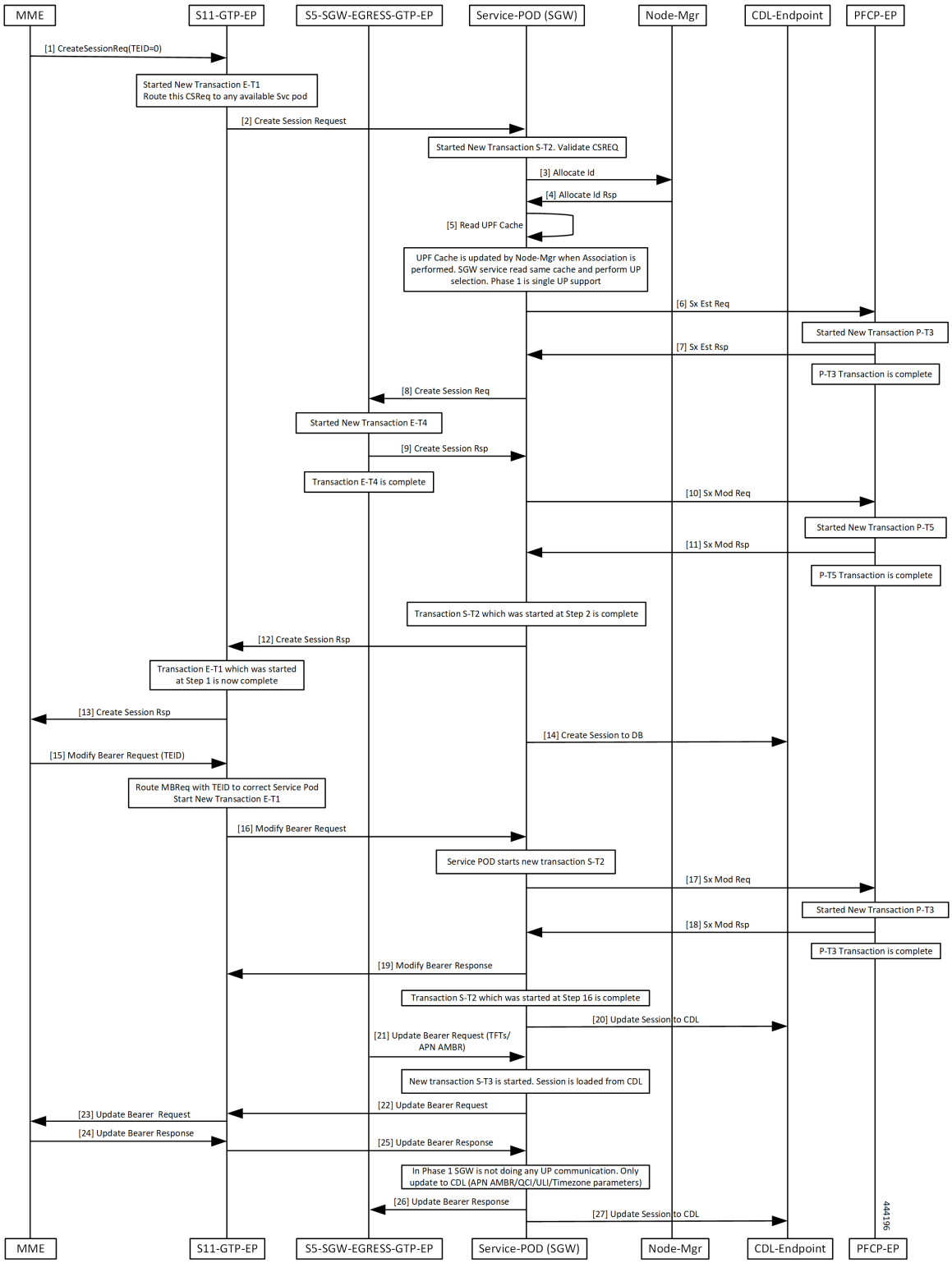
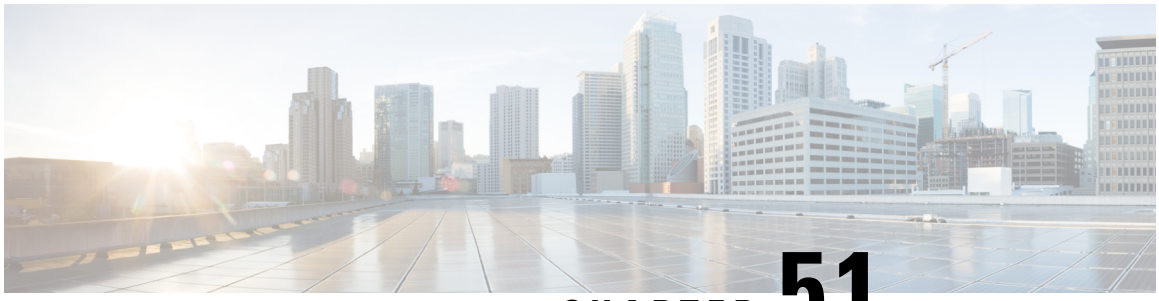


Table 253: Bearer Request/Response without UP Signaling Call Flow Description

Step	Description
1	MME sends Create Session Req to S11-GTP-EP with TEID value as zero.
2	Transaction E-T1 started. S11-GTP-EP sends Create Session Req to SGW-Service POD.
3, 4	Transaction S-T2 started. SGW-Service POD sends Allocate ID to Node-Mgr and receives response from it.
5, 6, 7	SGW-Service POD reads Node-Mgr updated UPF cache and performs UPF selection. SGW-Service POD sends Sx Est Req to PFCP-EP. Transaction P-T3 started. SGW-Service POD receives Sx Est Rsp from PFCP-EP.
8, 9	Transaction P-T3 completed. SGW-Service POD sends Create Session Req to S5-SGW-EGRESS-GTP-EP. Transaction E-T4 started. SGW-Service POD receives Create Session Rsp from S5-SGW-EGRESS-GTP-EP.
10, 11, 12	Transaction E-T4 completed. SGW-Service POD sends Sx Mod Req to PFCP-EP. Transaction P-T5 started. SGW-Service POD receives Sx Mod Rsp from PFCP-EP.
12	Transaction S-T2 ans P-T5 completed. SGW-Service POD sends Create Session Rsp to S11-GTP-EP.
13	Transaction E-T1 completed. S11-GTP-EP sends Create Session Rsp to MME.
14	SGW-Service POD sends Create Session to DB.
15	MME sends Modify Bearer Req with TEID value to S11-GTP-EP.
16	S11-GTP-EP routes MBReq with TEID to the exact SGW-Service POD. S11-GTP-EP sends Modify Bearer Request SGW-Service POD.
17	Transaction S-T2 completed. SGW-Service POD sends Sx Mod Req to PFCP-EP.
18	Transaction P-T3 started. SGW-Service POD receives Sx Mod Rsp from PFCP-EP.

Step	Description
19	Transaction P-T3 completed. SGW-Service POD sends Sx Mod Rsp to S11-GTP-EP.
20	Transaction S-T2 completed. SGW-Service POD sends Update Session to CDL.
21	S5-SGW-EGRESS-GTP-EP sends Update Bearer request with TFTs and APN AMBR to SGW-Service POD.
22, 23	Transaction S-T3 completed. SGW-Service POD sends Update Bearer Request to S11-GTP-EP. S11-GTP-EP forwards Update Bearer Request to MME.
24, 25	MME sends Update Bearer Rsp to S11-GTP-EP. S11-GTP-EP forwards Update Bearer Rsp to SGW-Service POD.
26	SGW-Service POD forwards Update Bearer Rsp to S5-SGW-EGRESS-GTP-EP.
27	SGW-Service POD sends Update Session to CDL.



CHAPTER 51

UPF Selection Support

- Feature Summary and Revision History, on page 705
- Feature Description, on page 706
- UPF Selection using DNN and DCNR Support, on page 706
- UPF Selection using Location Support, on page 710
- SGW-U selection based on system diagnostics, on page 715
- Combined UPF Selection for cnSGW-C and SMF, on page 716
- Block UPF to handle continuous Sxa session creation failures, on page 728
- UPF Selection OAM Support, on page 733

Feature Summary and Revision History

Summary Data

Table 254: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	UPF Selection using DCNR Support: Disabled – Configuration required to enable UPF Selection using DNN Support: Enabled – Always-on UPF Selection using Location Support: Disabled – Configuration required to enable Combined UPF Selection for cnSGW-C and SMF: Disabled – Configuration required to enable
Related Documentation	Not Applicable

Revision History

Table 255: Revision History

Revision Details	Release
Added support for UPF selection using Location. Added support for Combined UPF selection for cnSGW-C and SMF.	2021.02.0
First introduced.	2021.01.0

Feature Description

This feature describes the following UPF selection methods.

- DNN and DCNR
- Location support
- cnSGW-C and SMF to select same UPF instance

UPF Selection using DNN and DCNR Support

Feature Description

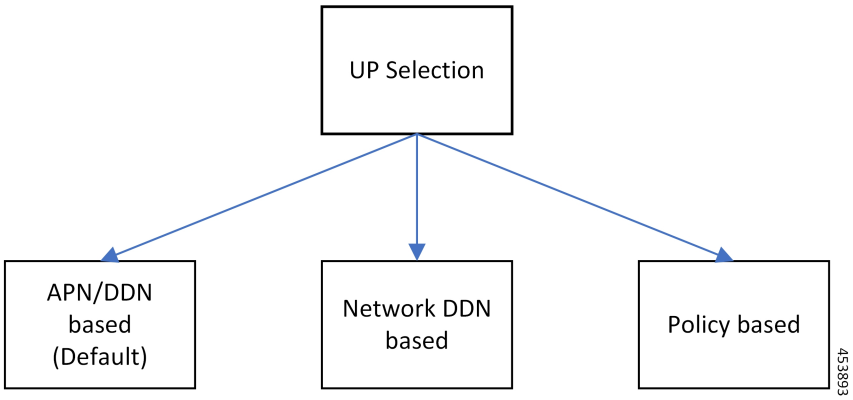
The following are the three UPF selection methods:

- DNN or APN based
- Network based
- Policy based

Note DNN is enabled when UPF selection policy isn't associated.

How it Works

This section describes how the feature works.



UPF Selection Methods

DNN or APN Based

- Create Session request message has APN information. This APN gets configured as part of DNN-list in the Network element profile for each user-plane.
- The PDN establishment considers these user-planes.
- UPF selection uses Capacity and Priority if many user-planes are available.

Network Based

- UPF selection considers DNN which got configured as part of the APN or DNN profile.
- This DNN is local SGW network specific DNN name.
- The same network DNN or APN name gets configured as part of DNN-list in the network element profile for each user-plane.
- Instead of using APN that comes in CSReq, local DNN is used for the UPF selection based on the DNN list.

  For example, in case of roaming scenario where APN is not known, this configuration helps in UPF selection.
- PDN establishment considers these user-planes.
- UPF selection uses Capacity and Priority if many user-planes are available.

UPF Selection Policy Based

- UPF selection profile configuration with parameters determines UPF for each precedence. The supported max number of precedencies are four.
- Each precedence parameter is a *Logical AND* condition. If DNN and DCNR are configured as precedence 1, then it searches for the DNN supported user-plane and enables DCNR based support. If this search criteria fails, it moves to the next (mostly 2) precedence and tries to evaluate that condition.
- UPF selection policy is associated with a DNN profile.

- UPF group provides characteristics to the network element profile which belongs to the same UPF group profile.
- UPF selection uses Capacity and Priority if many user-planes available.

Note
- cnSGW-C rejects the call with Create Session Response specifying cause as NO_RESOURCE_AVAILABLE when no UPF matches the precedence criteria.

Configuring UPF Selection Methods

This section describes how to configure the UPF selection methods.

Configuring UPF Group Profile-based UPF Selection

This section describes how to configure UPF group profile-based UPF selection.

Use the following commands to configure the UPF group profile-based UPF selection.

```
config
  profile upf-group upf_group_name
    dcnr [true | false]
    end
```

NOTES:

- **profile upf-group** *upf_group_name*—Specify the UPF group name. Must be a string.
- **dcnr [true | false]**—Specify to enable or disable support for dual connectivity with new radio. Default value is false.

Sample Configuration

Following is a sample configuration.

```
config
profile upf-group G1
 dcnr true
 end
```

Configuring Network-based UPF Selection

Use the following commands to configure the network-based UPF selection.

```
config
   profile network-element upfupf_name
      node-id  node_id_value
      n4-peer-address ipv4  ipv4_address
      n4-peer-address ipv6  ipv6_address
      n4-peer-port  port_number
      dual-stack-transport { true | false }
      upf-group-profile upf_group_profile_name
      dnn-list dnn_list
```

```
priority priority_value
capacity capacity_value
end
```

NOTES:

- **network-element upf** *upf_name*—Specify the UPF profile name.
- **node-id** *node_id_value*—Specify the Node ID of the UPF node.
- **n4-peer-address ipv4** *ipv4_address*—Specify the IPv4 address.
- **n4-peer-address ipv6** *ipv6_address*—Specify the IPv6 address.
- **dual-stack-transport { true | false }**—Enable the dual stack feature that allows you to specify an IPv6 or IPv4 address. Specify true to enable this feature.

  With this support, you can configure new eNBs and UPFs with IPv6 addresses for network expansion and continue with the existing eNBs and UPFs with IPv4 addresses for phased migration to IPv6 addresses.

Note Dual Stack (IPv4 and IPv6 IPs) S1-U UPF tunnel endpoint allocation is now supported during SGW relocation. This support is available with the 2026.03.3 release.

- **upf-group-profile** *upf_group_profile_name*—Specify the UPF group profile name.
- **dnn-list** *dnn_list*—Specify the DNN list supported by the UPF node.
- **priority** *priority_value*—Specify the static priority relative to other UPFs. This value is used for load balancing and must be an integer in the range of 0–65535. The default value is 1.
- **capacity** *capacity_value*—Specify the capacity relative to other UPFs. This value is used for load balancing and must be an integer in the range of 0–65535. The default value is 10.

Sample Configuration

The following is a sample configuration.

```
config
profile network-element upf UP1
 node-id      upf1@sgw.com
 upf-group-profile G1
 dnn-list [dnn1 dnn2]
 priority 20
 capacity 65535
 end
```

Configuring Policy based UPF Selection

This section describes how to configure Policy based UPF selection.

Use the following commands to configure the Policy based UPF Selection.

```
config
 policy upf-selection upf_selection_policyname
   precedence precedence_value location
```

```
  exit
  precedence precedence_value dnn
  exit
 exit
```

NOTES:

- **upf-selection** *upf_selection_policyname* - Specify the UPF selection policy name.
- **precedence** *precedence_value* - Specify the precedence for entry. Must be an integer in the range of 1-4.

Sample Configuration

Following is a sample configuration.

```
config
policy upf-selection upf_pol1
  precedence 1
    [ location ]
  exit
  precedence 2
    [ dnn ]
  exit
exit
```

Troubleshooting Information

This section describes the troubleshooting information that enables you to view the UPF selection using DNN and DCNR configuration issues.

Configuration Errors

```
show config-error | tab
ERROR COMPONENT   ERROR DESCRIPTION
-------------------------------------------------------------------------------------------------------
SGWProfile        Subscriber policy name : sub_policy in profile sgw1 is not configured
SubscriberPolicy  Operator policy : op_policy1 under subscriber policy sub_policy2 is not
configured
OperatorPolicy    Dnn policy name : dnn_policy1 in operator policy op_policy2 is not
configured
DnnPolicy        Dnn profile name : dnn_profile1 in dnn policy dnn_policy2 is not configured
DnnProfile        UPF selection policy name : upf_sel_policy1 in dnn profile dnn_profile2
is not configured
```

UPF Selection using Location Support

Feature Description

This feature supports Location-based UPF selection in Create Session Request message. It performs this selection as per the received TAI or ECGI or both TAI and ECGI values together.

Configuring the UPF Selection Feature

This section describes how to configure the UPF selection using location.

Configuring ECGI for EPS

This section describes how to configure ECGI for EPS.

New configuration and profile **ecgi-group** added to configure the list of individual ECGI values or the range of ECGI.

You can configure both ECGI list and ECGI range. ECGI range configuration is optional.

Use the following commands to configure the ECGI Configuration for EPS.

```
config
  profile ecgi-group ecgi_group_name
    mcc mcc_value
    mnc mnc_value
      ecgi list ecgi_list_name
      ecgi range start start_value end end_value
  exit
```

NOTES:

- **ecgi-group** *ecgi_group_name* - Specify the ECGI group name.
- **mcc** *mcc_value* - Specify the MCC value. Must be a three digit number. For example, 123
- **mnc** *mnc_value* - Specify the MNC value. Must be a two or three digit number. For example, 23 or 456
- **ecgi list** *ecgi_list* - Specify the list of ECGI values - 7 digit hex string Eutra Cell ID. For example, A12345f. Must be a string.
- **ecgi range start** *start_value* **end** *end_value* - Specify the ECGI range start and end values. Must be a string.

Note

- You can configure multiple ECGI range values.
- You can configure multiple [PLMN and ECGI values] under **ecgi-group** configuration.
- You can configure maximum of 16 PLMNs under **ecgi-group** configuration.
- You can configure maximum of 64 ECGI values in the ECGI list under a PLMN.
- Maximum defined number of ECGI ranges under a PLMN is 64.

Sample Configuration

Following is the sample configuration.

```
config
profile ecgi-group e1 mcc 123 mnc 45
ecgi list [ 1234567 abcdef0 ]
```

```
ecgi range start 1111111 end fffffff
exit
```

Verifying ECGI for EPS Configuration

This section describes how to verify the ECGI Configuration for EPS.

Use the following `show` command to view the ECGI configuration for EPS.

```
show running-config profile ecgi-group
profile ecgi-group e1
mcc 123 mnc 45
ecgi list [ 1234567 abcdef0 ]
ecgi range start 1111111 end fffffff
exit
exit
exit
```

Configuring TAI-Group

This section describes how to configure TAI-Group.

You can enhance the following TAI-Group configuration to support multiple TAI-Group configurations with different names.

Use the following commands to configure the TAI-Group.

```
config
  profile tai-group tai_group_name
    mcc mcc_value
    mnc mcc_value
      tac list tac_list
      tac range start start_value end end_value
  exit
```

NOTES:

- **tai-group** *tai_group_name* - Specify the TAI group name.
- **mcc** *mcc_value* - Specify the MCC value. Must be a three digit number. For example, 123
- **mnc** *mnc_value* - Specify the MNC value. Must be a two or three digit number. For example, 23 or 456
- **tac list** *tac_list* - Specify the list of TAC values - [0-9a-fA-F]{4}|[0-9a-fA-F]{6} - 4 digit or 6 digit hex string - Example A123, 1a2B3F. Must be a string.
- **tac range start** *start_value* **end** *end_value* - Specify the TAC range start and end values. Must be a string.

Note

- You can configure maximum of 16 PLMNs under a TAI-Group.
- You can configure maximum of 64 TAC values in a TAC list under a PLMN.
- Maximum defined number of TAC ranges under a PLMN is 64.

Sample Configuration

Following is the sample configuration.

```
config
profile tai-group TAI-GRP1
  mcc 123 mnc 234
    tac list [ 1a25 A123 ]
    tac range start B234 end b999
    exit
    tac range start C213 end c999
    exit
  exit
  mcc 231 mnc 45
    tac list [ 2a2B B123 ]
    tac range start d111 end d999
    exit
  exit
exit
```

Configuring Location-area-group

This section describes how to configure Location-area-group.

You can add new configuration and profile location-area-group. Configuration of **ecgi-group** and **tai-group** are optional.

Use the following commands to configure the Location-area-group.

```
config
   profile location-area-group location_area_group_name
      tai-group tai_group_name
      ecgi-group ecgi_group_name
   exit
```

NOTES:

- **location-area-group** *location_area_group_name* - Specify the location area group name.
- **tai-group** *tai_group_name* - Specify the TAI group name.
- **ecgi-group** *ecgi_group_name* - Specify the ECGI group name.

Sample Configuration

Following is the sample configuration.

```
config
profile location-area-group LOC_AREA_GRP_1
  tai-group TAI-AUTO-GRP1
  ecgi-group ECGI-AUTO-GRP1
exit
profile location-area-group LOC_AREA_GRP_2
  tai-group TAI-AUTO-GRP2
exit
```

Configuring UPF Group and UPF Selection Policy Enhancement

This section describes how to configure UPF Group and UPF Selection Policy Enhancement.

You can add new configuration under upf-group-profile to configure location-area-group-list.

Use the following commands to configure the UPF group anf UPF selection policy enhancement.

```
config
   profile upf-group upf_group_name
      location-area-group-list [area_group_list]
   exit
```

```
config
   policy upf-selection selection_policy_name
    precedence value  [ selection_parameter_list ]
   exit
```

NOTES:

- **upf-group** *upf_group_name* - Specify the UPF group name.
- **location-area-group-list** *area_group_list* - Specify the list of Location Area Group supported by UPF node.
- **upf-selection** *selection_policy_name* - Specify the UPF selection policy name.
- **precedence** *value* **[** *selection_parameter_list* **]** - Specify the precedence for entry. Must be an integer in the range of 1-4.



Note If pdn-type-subscription and pdn-type-session both are configured, pdn-type-subscription is considered.

Sample Configuration

Following is the sample configuration.

```
config
profile upf-group G1
 location-area-group-list [ LOC_AUTO_GRP_1 ]
exit
profile upf-group G2
 location-area-group-list [ LOC_AUTO_GRP_2 ]
exit
profile upf-group G3
 location-area-group-list [LOC_AUTO_GRP_1 LOC_AUTO_GRP_2 ]
exit
```

```
config
policy upf-selection upf_pol1
  precedence 1
    [ location ]
  exit
  precedence 2
    [ dnn ]
  exit
exit
```

SGW-U selection based on system diagnostics

This feature allows the network operator to configure a system-diagnostics configuration construct that allows operators to pin specific subscribers to a designated UPF, identified by its IP address. This capability is used for targeted call flow troubleshooting for specific subscribers.

When the cnSGWc receives a Create Session Request (CSR) from MME over the S11 interface for a subscriber, whose IMSI matches an entry in system-diagnostics SUPI configuration, cnSGW shall attempt to allocate the SGW-U identified by the configured preferred-up node-id.

Configure SUPI for SGW-U selection

Assign specific subscriber identities (SUPI) to a designated User Plane Function (UPF) node, ensuring targeted routing and improved control of subscriber traffic.

Use this task when you want to direct certain subscribers to a specific UPF node based on their identities.

Procedure

Step 1 Configure the SUPI value or a list of SUPI values in the System Diagnostics Configuration command mode.

Example:

```
Config
   system-diagnostics supisubscription_permanent_id
```

Step 2 Configure the preferred user plane IP address.

Example:

```
Config
   system-diagnostics supi subscription_permanent_id
     preferred-up node-id upf_address
     end
```

Step 3 Verify the configuration.

Example:

```
show running-config system-diagnostics supi
system-diagnostics supi [ 123456789012345 ]
   preferred-up node-id [ 209.165.200.230 209.165.200.236 ]
exit
```

Combined UPF Selection for cnSGW-C and SMF

Feature Description

This feature supports cnSGW-C and SMF to select the same UPF instance when the UPF and SMF are deployed on same cluster and UPF instance is available. If the UPF instance is not available, the UPF selection is based on the existing configurations.

Standards Compliance

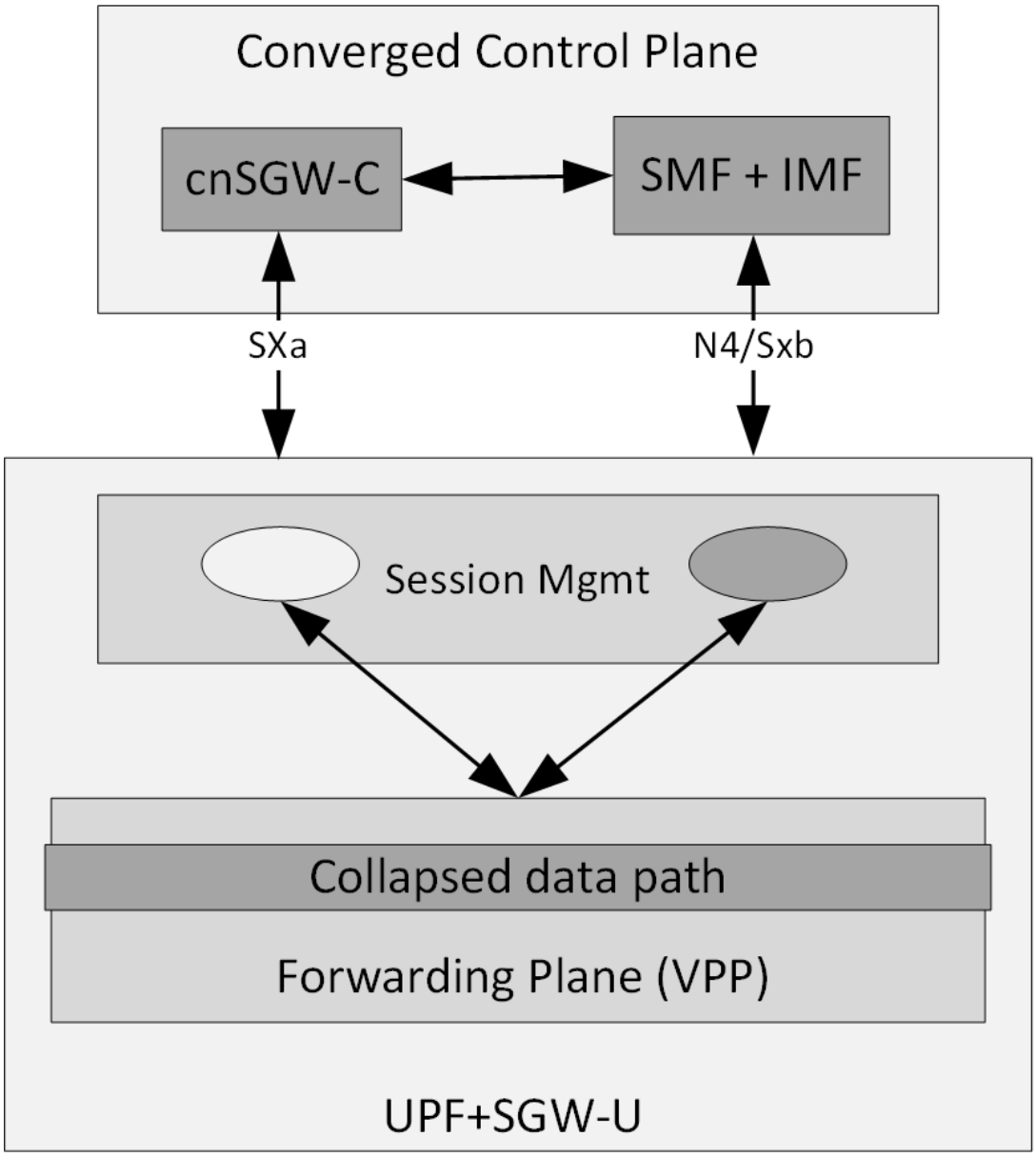
The Combined UPF Selection for cnSGW-C and SMF feature complies with the following standards:

- *3GPP TS 23.401*
- *3GPP TS 23.402*
- *3GPP TS 29.274*
- *3GPP TS 23.214*
- *3GPP TS 29.244*
- *3GPP TS 24.008*

How it Works

This section describes how this feature works.

System Architecture



cnSGW-C and SMF/IWF uses the same UPF instance, so that UPF can use those sessions to the collapsed data path.

Control plane (cnSGW-C and SMF) selects the same User-plane in various scenarios (initial attach, handover, and so on).

Following actions takes place during Initial Attach:

- cnSGW-C passes the SGW-U FQDN information of selected UPF instance to SMF in Initial attach.
- SMF selects the UPF instance as per the received SGW-U FQDN.

- Same UPF FQDN is configured at cnSGW-C and at SMF to create a correlation as part of the network element profile.

Call Flows

This section describes the key call flows of Combined UPF Selection for cnSGW-C and SMF feature.

Initial Attach on 4G for 5G Capable Device Call Flow

This section describes the Initial Attach on 4G for 5G Capable Device call flow.

Figure 134: Initial Attach on 4G for 5G Capable Device Call Flow

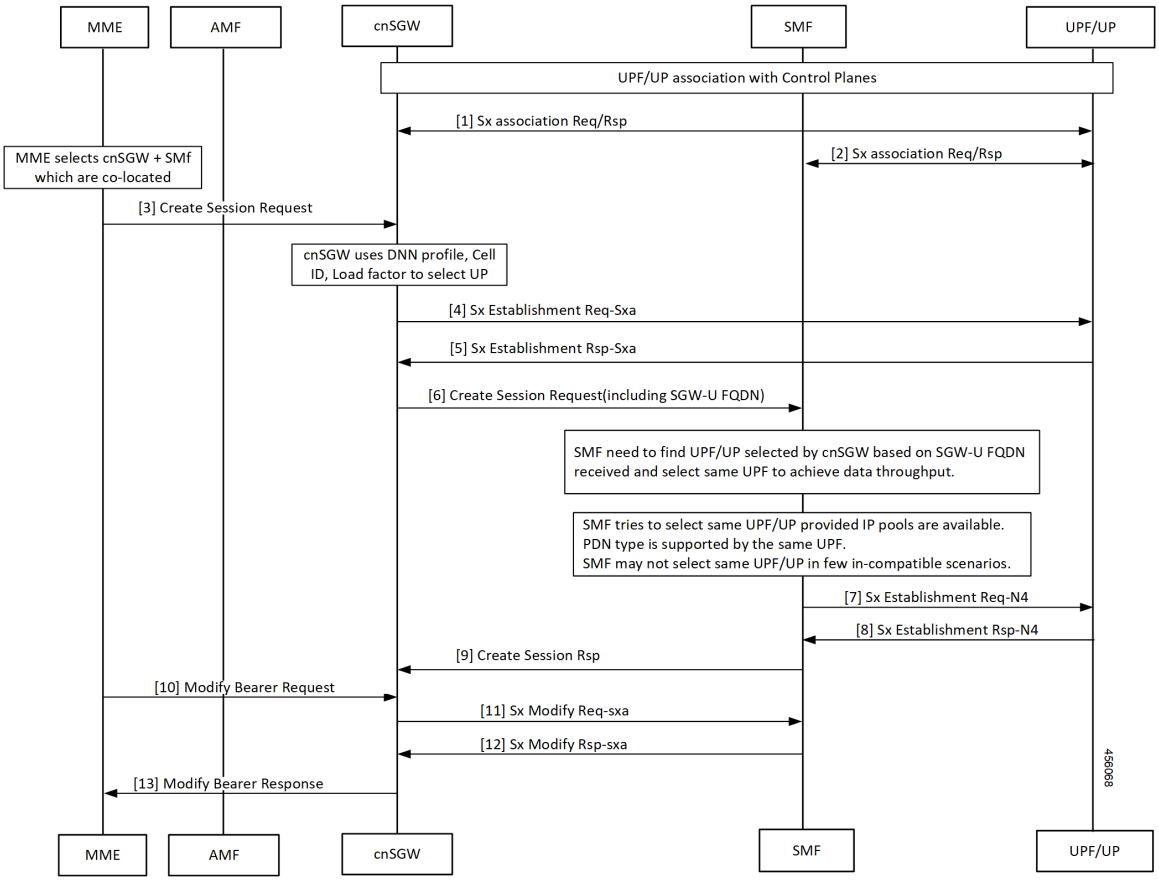


Table 256: Initial Attach on 4G for 5G Capable Device Call Flow Description

Step	Description
1	Established UPF association with control planes. cnSGW-C sends Sx Association Req/Rep to UPF.
2	SMF sends Sx Association Req/Rep to UPF.
3	MME sends Create Session Request to cnSGW-C after selecting co-located cnSGW-C and SMF.

Step	Description
4	cnSGW-C sends Sx Establishment Req (SXA) to UPF after selecting UPF using DNN profile, Cell ID, and local factors.
5	cnSGW-C receives Sx Establishment Res from UPF.
6	cnSGW-C sends Create Session Request to SMF including SGW-U FQDN.
7	SMF must find cnSGW-C selected UPF as per received SGW-U FQDN and select the same UPF to achieve data throughput. SMF tries to select same UPF when IP pools are available. Same UPF supports the PDN type. SMF may not select same UPF in few in-compatible scenarios. SMF send Sx Establishment Req N4 to UPF.
8	SMF receives Sx Establishment Res-N4 from UP.
9	cnSGW-C receives Create Session Response from SMF.
10	MME sends Modify Bearer Request to cnSGW-C.
11	cnSGW-C sends Sx Modify Req (SXA) to SMF.
12	cnSGW-C receives Sx Modify Res (SXA) to SMF.
13	cnSGW-C sends Modify Bearer Response to MME.

UPF Registration with User Plane ID Call Flow

This section describes the UPF Registration with User Plane ID call flow.

Figure 135: UPF Registration with User Plane ID Call Flow

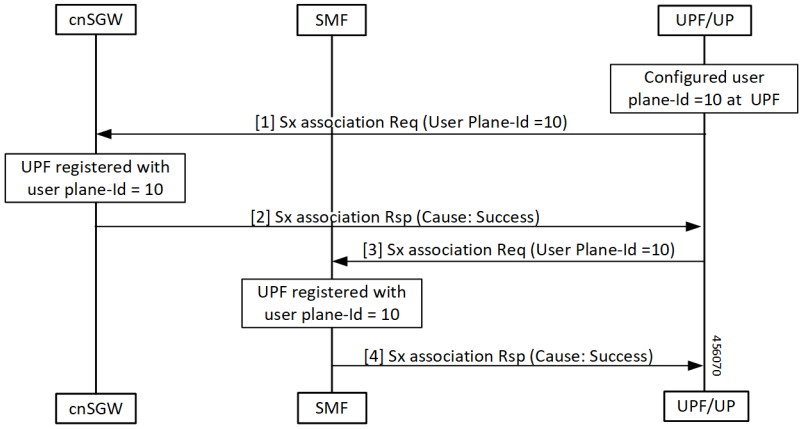


Table 257: UPF Registration with User Plane ID Call Flow Description

Step	Description
1	Configured User-plane ID at UPF cnSGW-C receives Sx association Request from UPF with configured User-plane ID.
2	UPF receives Sx association Response with Cause = SUCCESS from cnSGW-C.
3	SMF receives Sx association Request from UPF with configured User-plane ID.
4	UPF receives Sx association Response with Cause = SUCCESS from SMF.

Inter-SGW Handover on 4G RAT for 5G Capable Devices Call Flow

This section describes the Inter-SGW Handover on 4G RAT for 5G Capable Devices call flow.

Figure 136: Inter-SGW Handover on 4G RAT for 5G Capable Devices Call Flow

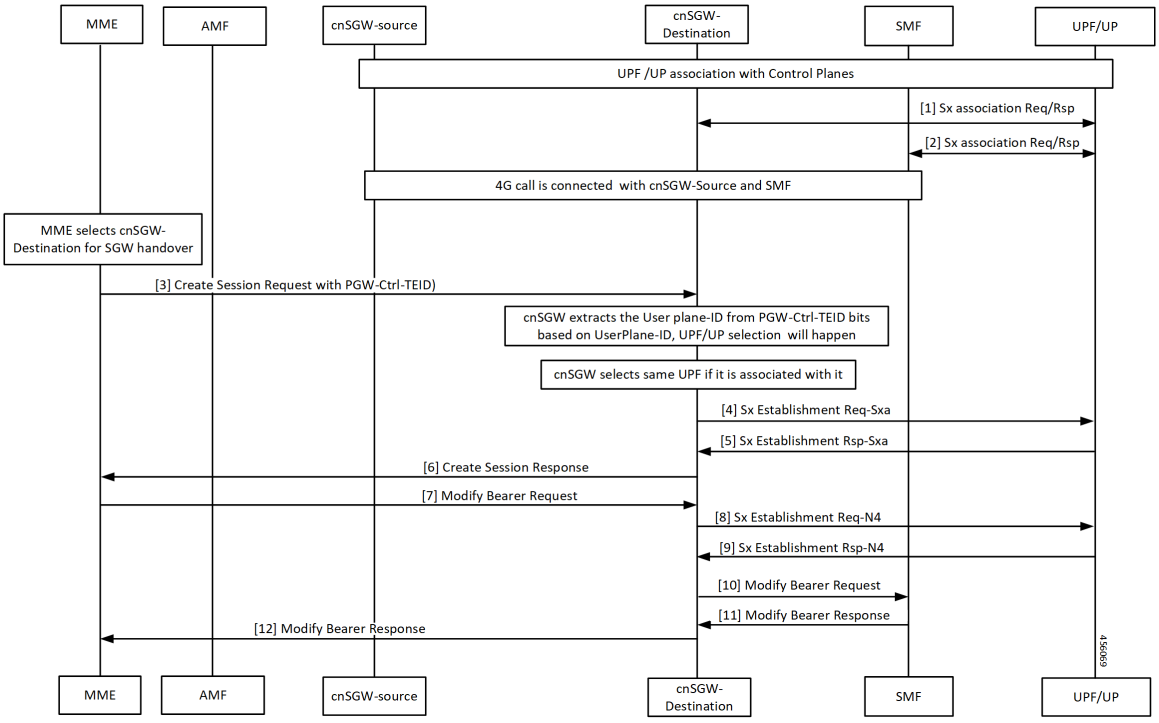


Table 258: Inter-SGW Handover on 4G RAT for 5G Capable Devices Call Flow Description

Step	Description
1	Established UPF association with control planes. UPF sends Sx Association Req/Rep to destination cnSGW-C.
2	UPF sends Sx Association Req/Rep to SMF.

Step	Description
3	4G call connected between cnSGW-C-source and SMF. MME selects cnSGW-C-Destination for the SGW handover. MME sends Create Session Req with PGW-Ctrl-TEID to cnSGW-C-Destination.
4	cnSGW-C extracts same associated UPF ID from PGW-Ctrl-TEID. cnSGW-C-Destination sends Sx Establishment Req (SXA) to UPF.
5	cnSGW-C-Destination receives Sx Establishment Rsp (SXA) from UPF.
6	cnSGW-C-Destination sends Create Session Response to MME.
7	cnSGW-C-Destination receives Modify Bearer Request from MME.
8	cnSGW-C-Destination sends Sx Establishment Req-N4 to UPF.
9	cnSGW-C-Destination receives Sx Establishment Rsp-N4 from UPF.
10	cnSGW-C-Destination sends Modify Bearer Request to SMF.
11	cnSGW-C-Destination receives Modify Bearer Response from SMF.
12	cnSGW-C-Destination forwards Modify Bearer Request to MME.

5G to EPS Handover Using N26 Interface – cnSGW-C and SMF Separate Node Call Flow

This section describes the 5G to EPS Handover Using N26 Interface – cnSGW-C and SMF Separate Node call flow.

Figure 137: 5G to EPS Handover Using N26 Interface – cnSGW-C and SMF Separate Node Call Flow

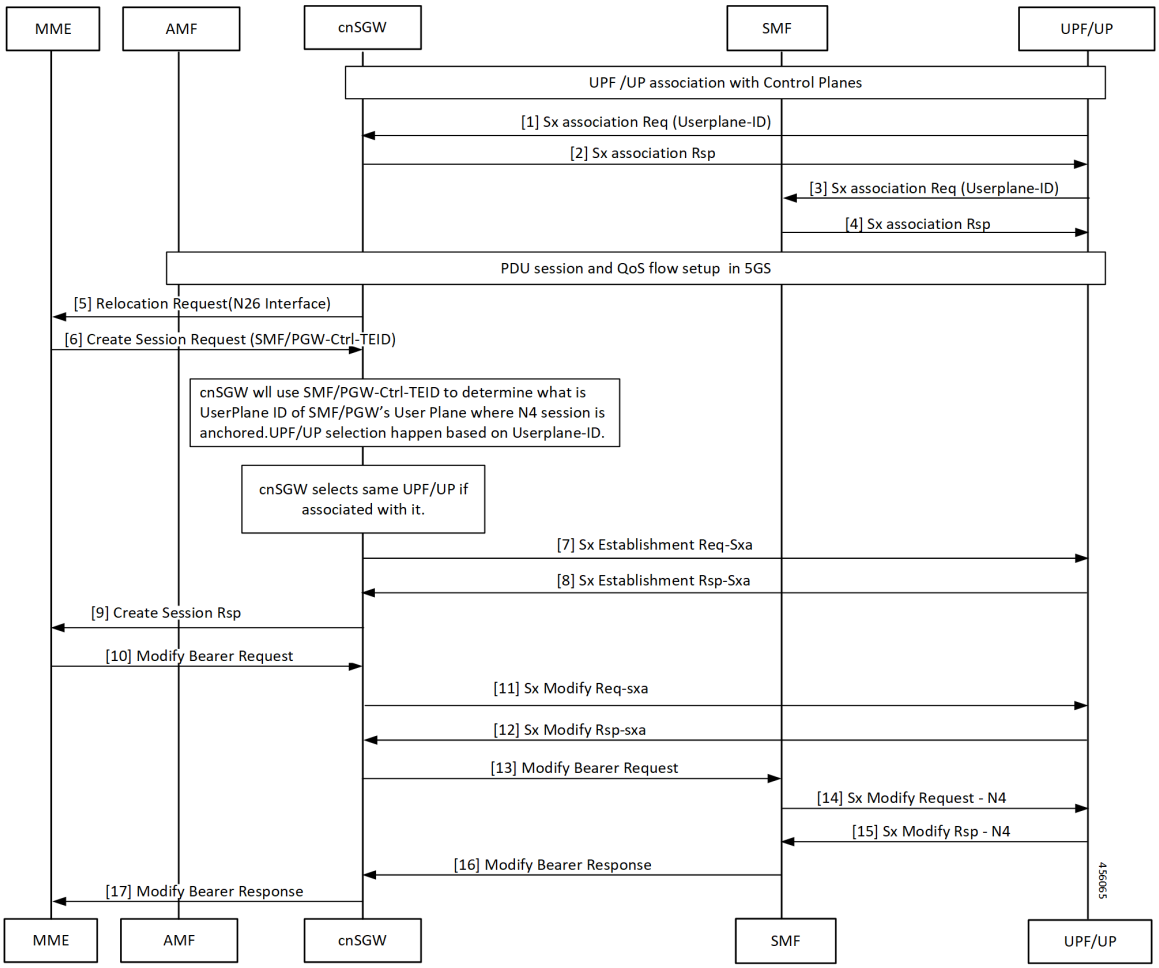


Table 259: 5G to EPS Handover Using N26 Interface – cnSGW-C and SMF Separate Node Call Flow Description

Step	Description
1	cnSGW-C selects associated same UPF. cnSGW-C receives Sx Establishment Request with User-plane ID from UPF/UP.
2	cnSGW-C sends Sx Establishment Response from UPF/UP.
3	UPF/UP sends Sx Association Request with User-plane ID to SMF.
4	SMF sends Sx Association Response to UPF/UP.
5	cnSGW-C sends Relocation Request to MME on interface N26.
6	MME sends Create Session Request to cnSGW-C with SMF and PGW-ctrl-TEID information.
7	cnSGW-C selects associated same UPF. cnSGW-C sends Sx Establishment Req (SXA) to UPF/UP.

Step	Description
8	cnSGW-C receives Sx Establishment Rsp (SXA) from UPF/UP.
9	cnSGW-C sends Create Session response to MME.
10	MME sends Modify Bearer Request to cnSGW-C.
11	cnSGW-C sends Sx Modify Req (SXA) to UPF/UP.
12	cnSGW-C receives Sx Modify Rsp (SXA) from UPF/UP.
13	cnSGW-C sends Modify Bearer Request to SMF.
14	SMF sends Sx Modify Request – N4 to UPF/UP.
15	SMF receives Sx Modify Response from UPF/UP.
16	cnSGW-C receives Modify Bearer Response to SMF.
17	cnSGW-C forwards Modify Bearer Response to MME.

Wi-Fi to LTE Handover Call Flow

This section describes the Wi-Fi to LTE Handover call flow.

Figure 138: Wi-Fi to LTE Handover Call Flow

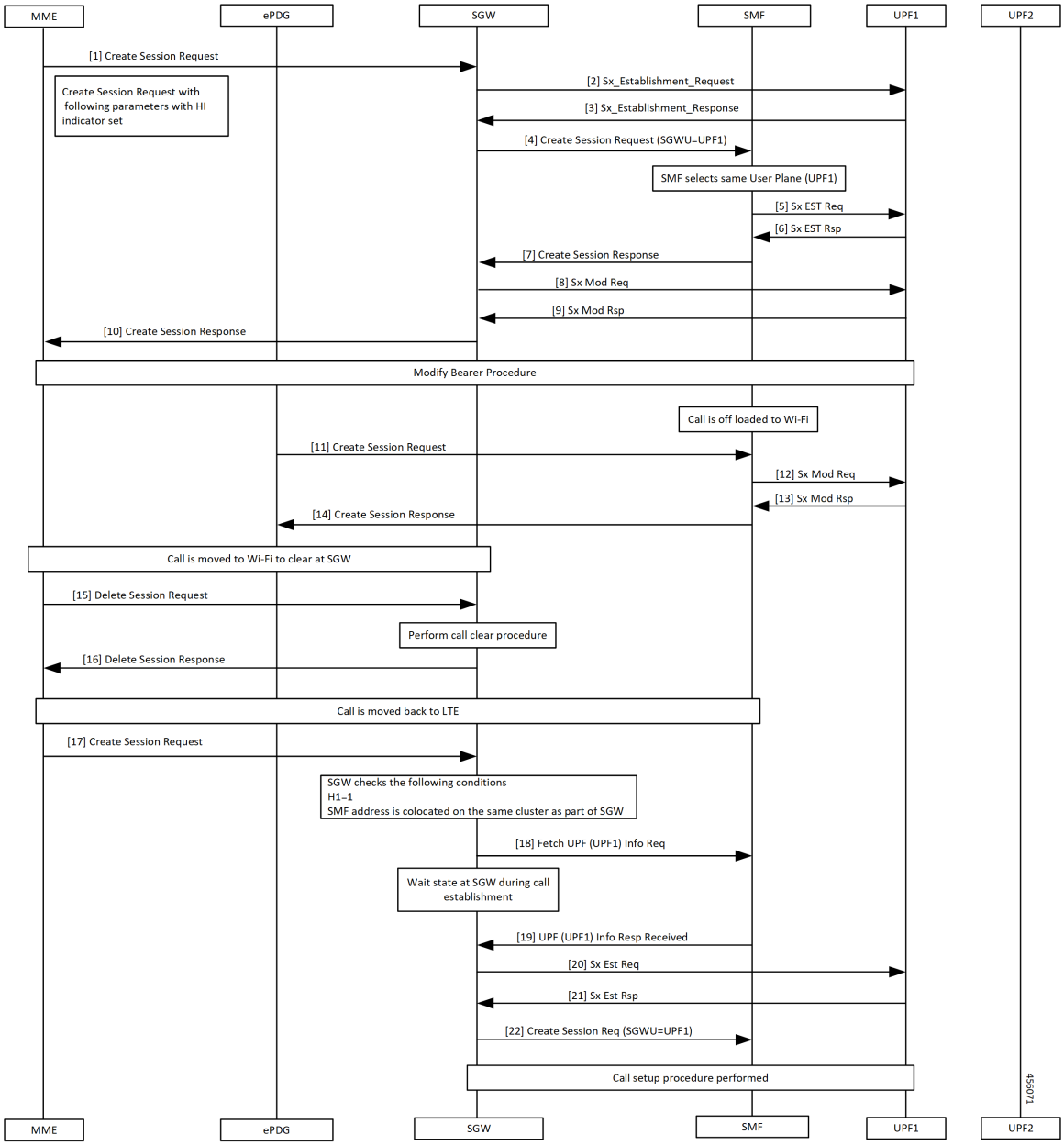


Table 260: Wi-Fi to LTE Handover Call Flow Description

Step	Description
1	MME sends create Session Request to SGW.
2	SGW sends Sx Establishment Request to UPF1.
3	SGW receives Sx Establishment Response from SMF.
4	SGW sends Create Session Request to SMF. SMF selects same UPF.

Step	Description
5	SMF sends Sx Establishment Request to UPF1.
6	SMF receives Sx Establishment Response from UPF1.
7	SMF sends Create Session Response to SGW.
8	SGW sends Sx Mod Request to UPF1.
9, 10	SGW receives Sx Mod Response from UPF1 and forwards to MME.
11	Modify Bearer procedure takes place and off-loaded call to Wi-Fi. ePDG sends Create Session Request to SGW.
12	SMF sends Sx Mod Request to UPF1.
13	SMF receives Sx Mod Response from UPF1.
14	ePDG receives Create Session Response from SGW.
15	Call moved to Wi-Fi to clear at SGW. MME sends Delete Session Request to SGW.
16	MME receives Delete Session Response from SGW after performing call clear procedure.
17	MME sends Create Session Request to SGW.
18	SGW sends fetch UPF (UPF1) info request to SMF after checking SMF as same cluster as cnSGW-C.
19	SGW receives UPF (UPF1) Info Response from SMF.
20	SGW sends Sx Establishment Request to UPF1.
21	SGW receives Sx Establishment Response from UPF1.
22	SGW sends Create Session Request with SGWU=UPF1 to UPF1 and performs call setup procedure.

Configuring the Combined UPF Selection for cnSGW-C and SMF

This section describes how to configure the Combined UPF Selection for cnSGW-C and SMF.

Configuring Converged-Core Profile

This section describes how to configure the Converged-Core Profile.

Use the following commands to configure the profile converged-core with UPF selection enabled.

```
config
  profile converged-core core_name
    max-upf-index value
    no upf-selection disable
  exit
```

Use the following commands to configure the profile converged-core with UPF selection disabled.

```
config
  profile converged-core core_name
    max-upf-index value
    upf-selection disable
  exit
```

NOTES:

- **converged-core** *core_name* - Specify the converged core prolile name.
- **max-upf-index** *value* - Specify the UPF index value. Must be an integer in the range of 1-1023.
- **no upf-selection disable** - Enable colocated UPF selection.
- **upf-selection disable**- Disable colocated UPF slection.

Sample configuration

Following is a sample configuration with UPF selection enabled.

```
config
profile converged-core cc1
max-upf-index 1023
no upf-selection disable
exit
```

Following is a sample configuration with UPF selection disabled.

```
config
profile converged-core cc1
max-upf-index 1023
upf-selection disable
exit
```

Verifying the Profile Converged-core Configuration

This section describes how to verify the Profile Converged-core configuration.

Use the following show command to view the Profile Converged-Core configuration with UPF selection enabled.

```
show running-config profile converged-core cc1
profile converged-core cc1
max-upf-index 1023
no upf-selection disable
exit
```

Use the following show command to view the Profile Converged-Core configuration with UPF selection disabled.

```
show running-config profile converged-core cc1
profile converged-core cc1
max-upf-index 1023
upf-selection disable
exit
```

Configuring Node-ID

This section describes how to configure the Node-ID.

Use the following commands to configure the Node-ID.

```
config
  profile network-element upf upf_name
     node-id node_id_value
  exit
```

NOTES:

- **network-element upf** *upf_name* - Specify the UPF peer network element name.
- **node-id** *node_id_value* - Specify the Node ID of the UPF node. Must be a string

Sample Configuration

Following is a sample configuration.

```
config
profile network-element upf upf1
node-id upf1@cn.com
exit
```

Block UPF to handle continuous Sxa session creation failures

Table 261: Feature History

Feature Name	Release Information	Description
Blocking UPF to handle Sxa session establishment failures	2025.02.0	This feature allows the cnSGWc to blocklist a specific UPF node when it continuously rejects session establishment requests. The UPF may reject these requests from the cnSGWc due to various errors, such as licensing issues. However, without information on the issues at the UPF, the cnSGWc might repeatedly select the same UPF, leading to an increased number of session creation failures. By enabling this feature, the network operators can prevent the cnSGWc from repeatedly selecting the same UPF for a defined time interval. **Commands Introduced:** • **[no] activated-features upf-blocklisting [use-alert** *custom_rule_name* **]**: This CLI is configured under converged core profile configuration mode to enable UPF blocklisting feature on cnSGWc. • **reactivate-peer condition [blocked] attributes [frequency** *frequency_timer* **]**: This CLI is configured under the endpoint configuration mode to reactivate blocked UPF after a frequency timer is over. **Default Setting:** Disabled—Configuration Required to Enable

During the Sxa session creation process, the cnSGWc selects a UPF based on configured options and parameters received in the create request. However, the selected UPF might occasionally be unable to handle the requests due to various issues, such as licensing problems, leading to repeated rejection of session establishment requests.

Since the cnSGWc lacks information about the issues at the UPF, it may continue to select the same UPF, resulting in repeated session creation failures.

This feature allows network operators to blocklist a UPF node by configuring alerts that trigger when the Sxa session establishment failure rate exceeds a defined threshold. It also enables the configuration of a time interval during which the specified UPF node remains blocklisted and allows for reactivating the blocked UPF once the timer expires.

How handling continuous Sxa session creation failures works

The network operator configures the alerts in the Alert Manager with rule name, procedure type, status, interface type, and other parameters defining the conditions for triggering.

Here is a sample of alert configuration:

```
alerts rules group PeerFailure
    rule upf_inactive
    expression "(sum by (namespace,upf_id,gr_instance_id)
    (proto_pfcp_msg_stats{status='failed'})) >= 1000”
    duration   5m
    severity   major
    type       "Communications Alarm"
    annotation summary
    value "NF=UPF Value={{ $value }} - UPF inactive"
    exit
   exit
exit
```



Note The rule name specified in the configuration for enabling UPF blockin activation configuration should be same as the rule name configured in the alert.

The Alert Manager informs the cnSGWc, when the session creation failure reaches a threshold limit. If UPF blocklisting and reactivation configurations are enabled, cnSGWc blocks the specific UPF for the defined frequency timer, and looks for an alternate UPF for session creation.

For more information on UPF blosklisting and UPF reactivation, see the Configure cnSGWc to handle continuous Sxa session creation failures topic.

Once the defined frequency timer expires, cnSGWc removes that UPF from the blocked status and considers it for session creation.

Configure cnSGWc to handle continuous Sxa session creation failures

Perform these tasks to resolve the continuous Sxa session creation failures:

- Enable UPF blocking
- Reactivate the blocked UPF

Enable UPF blocking

This task allows the operator to enable or disable the UPF blocklisting functionality. Also, it allows the operator to configure an optional custom rule name to be used for subscribing the alerts for UPF blocklisting and notification from Event Manager.

Follow these steps to enable UPF blocklisting:

Procedure

Step 1 Use the command **profile converged-core** *cc_profile_name* to enter the converged-core profile configuration mode.

Example:

```
[sgw] sgw# config
[sgw] sgw(config)# profile converged-core ccg1
[sgw] sgw(config-converged-core-ccg1)#
```

Step 2 Use the command **[no] activated-features upf-blocklisting [use-alert** *custom_rule_name* **]** to enable the UPF blocklisting and setting a custom rule name.

Example:

```
[sgw] sgw(config-converged-core-ccg1)# activated-features upf-blocklisting use-alert
upf_inactive1
[sgw] sgw(config-converged-core-ccg1)#
```

The **[use-alert** *custom_rule_name* **]** is an optional CLI. It allows the network operator to configure a custom rule name for alert subscriptions.

Note

- The default value of the command **[use-alert** *custom_rule_name* **]**, is "upf_inactive" for the CLI **activated-features upf-blocklisting**, if the **use-alert** is not configured with a custom rule name.
- Changing the custom rule name during run-time requires the previous rule name to be removed. Configuring the new custom rule name before removing the previous rule name, results in activating both the rule names.
- The new alert configuration parameters should be configured with the new rule name in CEE alert configuration (CEE ops-center).

Step 3 Use the command **exit** to save and exit the converged core profile configuration mode.

Example:

```
[sgw] sgw(config-converged-core-ccg1)# exit
[sgw] sgw(config)#
```

This task enables blocklisting of a defined UPF.

Reactivate blocked UPF

This task allows the network operator to configure the frequency interval during which the specific UPF is marked as blocked for selection. Once the frequency interval ends, the UPF becomes available for selection.

Follow these steps to reactivate the blocklisted UPF:

Before you begin

Before performing this task, you must enable UPF blocking.

Procedure

Step 1 Use the command **instance instance-id***gr_instance_id* to create an instance of the instance profile.

Example:

```
[sgw] sgw# config
[sgw] sgw(config)# instance instance-id 1
[sgw] sgw(config-instance-id-1)#
```

Step 2 Use the command **endpoint pfcp** to enter the endpoint configuration mode.

Example:

```
[sgw] sgw(config-instance-id-1)# endpoint pfcp
[sgw] sgw(config-endpoint-pfcp)#
```

Step 3 Use the command **interface** *interface_type*to configure the parameters for Sxa interface.

Example:

```
[sgw] sgw(config-endpoint-pfcp)# interface sxa
[sgw] sgw(config-interface-sxa)#
```

Step 4 Use the command **reactivate-peer condition [blocked] attributes [frequency** *frequency_timer* **]** to reactivate the blocked UPF once the timer is over.

Example:

```
[sgw] sgw(config-interface-sxa)# reactivate-peer condition blocked attributes frequency
 60
[sgw] sgw(config-interface-sxa)#
```

- The value of the command **frequency** ranges between 60 to 86400.
- The frequency parameter does not have any default values.
- It is mandatory to configure the frequency parameter within the defined range. If the frequency parameter is not configured within the range, the system by default takes 0 as the input. As a result, the identified UPF does not get blocklisted even if the alarm is triggered.

Step 5 Use the command **exit** to save and exit the interface configuration mode.

Example:

```
[sgw] sgw(config-interface-sxa)# exit
[sgw] sgw(config-endpoint-pfcp)#
```

Monitoring and troubleshooting

This section discusses the bulkstatistics and show commands used for monitoring and troubleshooting this feature.

Show command and output

The show command **show userplane all** displays the current status of UPF under the NodeStatus parameter. The new state **BLOCKED (5)**indicates that the specific UPF's current status is blocked.

Here is an example of the show command output:

```
[sgw] sgw#show userplane all
{
  "10.1.46.72:10.1.47.208": {
    "NodeIdType": 1,
    "NodeId": "10.1.46.72",
    "NodePort": 8805,
    "NodeStatus": 5,
    "Capacity": 65535,
    "Priority": 10,
    "DnnList": [
      "ims1",
      "ims2",
      "intershat"
    ],
    "PrimaryNodeMgrInst": {
      "IsActive": true
    },
    "PeerNodeMgrInst": {
      "InstanceId": 1,
      "IsActive": true
    },
    "EpIp": "10.1.47.208",
    "EpPort": 8805,
    "UpEpKey": "10.1.46.72:10.1.47.208",
    "recoveryInfo": {
      "SvcRecoveryTime": 3948948938,
      "PeerRecoveryTime": 3948786054
    },
    "ConnectedTime": 3948949630,
    "IntfType": 4,
    "UpProfName": "upf1",
    "OverloadTimer": {},
    "NegotiatedCPFeatures": 2147483648,
    "IsAssocUpdSuccess": true,
    "UpfInactiveValidityTime": 1739960992
  }
```

Bulkstats

These statistics are introduced as part of this feature:

Statistics	Description	Label(s)
prometheus_alert_received_total	Total number of valid Prometheus alerts received on event manager on its rest ep.	alert_rule
prometheus_alert_invalid_data_total	Total number of Prometheus alerts with invalid data received on event manager on its rest ep.	err_code
pubsub_subscriptions_total	Total number of subscription for producer pod.	event_name, consumer_instance

Statistics	Description	Label(s)
pubsub_events_produced_total	Total number of event published to subscribed pod.	event_name, instance, retry, retry_attempt, cause
pubsub_events_produced_ack_total	Total number of event published status to subscribed pod.	event_name, consumer_instance, status, retry, retry_attempt
pubsub_events_consumed_ack_total	Total number of event publish ack to producer.	event_name, dest_host, retry, retry_attempt
proto_pfcp_msg_stats	The number of failures observed during session establishment procedure.	app_name, cause, cluster, data_center, gr_instance_id, instance_id, interface_type, message_name,upf_id, service_name, status, trans_type

Note The statistics **proto_pfcp_msg_stats** counter increases only when the UPF blocklisting feature is enabled under converged core profile configuration activated-feature list. It is mandatory to specify the "**upf_id**" under the list of labels grouped by as part of the expression to make this feature work. This parameter helps identify the UPF, which needs to be blocklisted. Without this, the functionality does not work as expected.

UPF Selection OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

UE Disconnect Statistics

```
sgw_ue_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
reason="userplane_info_not_available",service_name="sgw-service"} 24
```

PDN Disconnect Statistics

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
pdn_type="ipv4",rat_type="EUTRAN",reason="userplane_info_not_available",service_name="sgw-service"}
 8
```

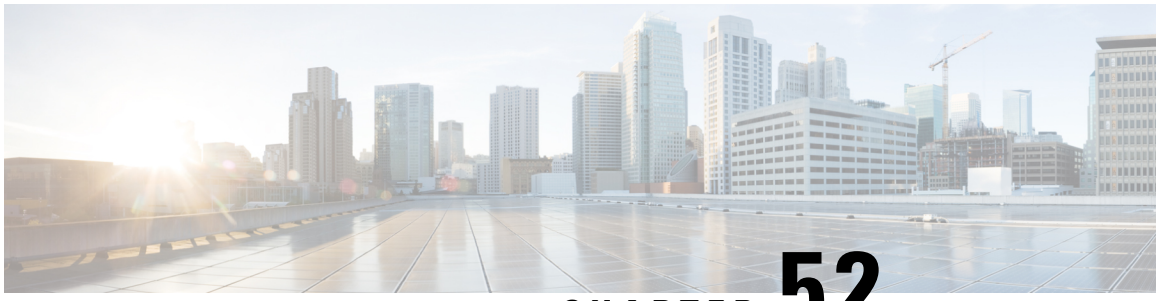
```
sgw_pdn_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
pdn_type="ipv4v6",rat_type="EUTRAN",reason="userplane_info_not_available",service_name="sgw-service"}
 15
```

```
sgw_pdn_disconnect_stats{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
pdn_type="ipv6",rat_type="EUTRAN",reason="userplane_info_not_available",service_name="sgw-service"}
 1
```

SGW Service Statistics

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="userplane_selection_fail",
instance_id="0",interface="interface_sgw_ingress",reject_cause="no_resources_available",service_name=
"sgw-service",sgw_procedure_type="initial_attach",status="failure",sub_fail_reason=""}
 22
```

```
sgw_service_stats{app_name="smf",cluster="cn",data_center="cn",fail_reason="userplane_selection_fail",
instance_id="0",interface="interface_sgw_ingress",reject_cause="no_resources_available",service_name="sgw-service",
sgw_procedure_type="secondary_pdn_creation",status="failure",sub_fail_reason=""}
 2
```



CHAPTER 52

VoLTE Call Prioritization

- Feature Summary and Revision History, on page 735
- Feature Description, on page 736
- How it Works, on page 736
- Feature Configuration, on page 736
- OAM Support, on page 739

Feature Summary and Revision History

Summary Data

Table 262: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Always-on
Related Documentation	Not Applicable

Revision History

Table 263: Revision History

Revision Details	Release
Added support for Extended and Non-Standard (Operator-defined) QCI Values.	2021.02.3
First introduced.	2021.02.0

Feature Description

cnSGW-C provides:

- CLI support to mark QCI as IMS media.
- CLI support to display whether session/bearer is VoLTE or not in `show subscriber` output.
- Counter support to identify number of VoLTE subscribers in the system.
- Sx message priority configuration based on VoLTE marked session.

How it Works

This section describes how this feature works.

- SGW profile represents SGW service.
- SGW profile has associated subscriber policy, which helps to select the Operator Policy.
- Operator Policy has DNN policy associated with it.
- DNN policy has DNN profile associated with it which has the QCI mark for marking VoLTE subscriber for priority.

Based on the QCI marking as IMS, *volteBearer* and *volteSession* flags are set internally when you execute `show subscriber` command.

- *volteBearer* is a bearer level flag. If bearer QCI is present in marked QCI list, *volteBearer* flag is set as true and the bearer is considered as **volteBearer**.
- *volteSession* is a session level flag. This flag is set as **true** if there's a VoLTE bearer present in any PDN of that subscriber.

Feature Configuration

Configuring this feature involves the following steps:

- Configure the call priority. For more information, refer to Configuring the Priority, on page 736.
- Configure the message priority. For more information, refer to Sx Message Priority, on page 739.

Configuring the Priority

This section describes how to configure the priority.

CLI is used to mark the QCI level as VoLTE media under dnn profile. If requested QCI in the call matches with the marked QCI, SGW sets the *volteSession* and *volteBearer* flags. If a subscriber session has **volteSession**, then that subscriber has the highest priority compared to other subscribers.

```
config
  profile dnn profile_name ims mark qci qci_value
```

NOTES:

- **profile dnn** *profile_name*—Specify the DNN profile name.
- **mark**—For marking standard QCI value as IMS media.
- **qci** *qci_value*—Specify the QCI value. The following QoS Class Identifiers are supported:

  Standard: 1-9

  Extended: 65, 66, 69, 70, 80, 82, 83

  Non-Standard (Operator-defined): 128-254

Configuration Example

The following is an example configuration.

```
config
   profile dnn dnn1 ims mark qci [ 2 3 4 ]
   end
```

Configuration Verification

To verify the configuration

```
show full-configuration profile dnn dnn1
profile dnn dnn1
 ims mark qci [ 2 3 4 ]
```

Based on the QCI marking as IMS, *volteSession* and *volteBearer* flags are set internally when you execute `show subscriber` command.

This section provides sample output.

```
show subscriber namespace sgw imsi 123456789012348
subscriber-details
{
  "subResponses": [
    {
      "status": true,
      "genericInfo": {
        "imsi": "imsi-123456789012348",
        "mei": "imei-123456786666660",
        "msisdn": "msisdn-223310101010101",
        "accessType": "EUTRAN",
        "plmnId": {
          "mcc": "123",
          "mnc": "456"
        },
        "sgwProfileName": "sgw1",
        "unAuthenticatedImsi": "No"
      },
      "s11cInterfaceInfo": {
        "sgwTeid": "[0x12000147] 301990215",
        "sgwIPv4Address": "209.165.201.19",
        "mmeTeid": "[0x62b5] 25269",
        "mmeIPv4Address": "209.165.201.20"
      },
```

```
"pdnInfoList": {
  "totalPdn": 1,
  "pdnInfo": [
    {
      "pdnId": "PDN-1",
      "apn": "intershat",
      "attachType": "Initial Attach",
      "sgwRelocState": "N/A",
      "operatorPolicyName": "N/A",
      "dnnProfileName": "N/A",
      "defaultEbi": 5,
      "pdnType": "IPv4",
      "allocatedIPv4": "209.165.201.26",
      "apnSelectionMode": "Subscribed",
      "ambrUplink": "10 Kbps",
      "ambrDownlink": "20 Kbps",
      "s5cInterfaceInfo": {
        "sgwTeid": "[0x52000147] 1375732039",
        "sgwIPv4Address": "209.165.201.19",
        "pgwTeid": "[0x339a] 13210",
        "pgwIPv4Address": "209.165.201.18"
      },
      "sxaInterfaceInfo": {
        "selectedUP": "209.165.201.20",
        "upEpKey": "209.165.201.20:209.165.201.19",
        "cpSeid": "[0x1200014752000147] 1297038098512740679",
        "upSeid": "[0x2712] 10002"
      },
      "bearerInfoList": {
        "totalBearer": 1,
        "bearerInfo": [
          {
            "bearerId": "Bearer-1",
            "state": "Connected",
            "ebi": 5,
            "isDefaultBearer": true,
            "qosInfo": {
              "qci": 6,
              "arp": 113
            },
            "s1uInterfaceInfo": {
              "sgwTeid": "[0x62b7] 25271",
              "sgwIPv4Address": "209.165.200.226",
              "eNodeBTeid": "[0x62b8] 25272",
              "eNodeBIPv4Address": "209.165.201.20"
            },
            "s5uInterfaceInfo": {
              "sgwTeid": "[0x62b6] 25270",
              "sgwIPv4Address": "209.165.201.1",
              "pgwTeid": "[0x339b] 13211",
              "pgwIPv4Address": "209.165.201.18"
            },
            "chargingId": 303174163
          }
        ]
      },
      "uli": {
        "mcc": "123",
        "mnc": "456",
        "tac": "0x92a",
        "eci": "0x12d687"
      },
      "uetimeZone": {
        "timeZone": "+0:15",
```

```
            "dayLightSavingTime": "+1 hour"
          },
          "plmnType": "VISITOR"
        }
      ]
    }
  }
 ]
}
```

Sx Message Priority

This section describes the Sx message priority.

Based on the VoLTE flags (volteSession and volteBearer), SGW sets the message priority in Sx request messages (Sx-Est, Sx-Mod, and so on) while processing the received requests/responses (for example, S5-CBReq, S11CBResp, S11-UBRes, S5DBReq, and so on).

Sx Message Priority is set when the session is marked for volteSession (or it has VoLTE QCI).

In Sx Req (Establishment or Modification) Message Header:

```
"HEADERS": {
      "length": 30,
      "msg_type": 52,
      "priority": 2,        <<< VOLTE Session priority value
      "priority_flag": 1,   <<< Priority flag is true i.e 1
      "retransmit": 0,
      "seid": 10002,
      "seid_flag": 1,
      "seq_number": 22,
      "version": 1
    },
```

OAM Support

This section describes operations, administration, and maintenance information for this feature.

Bulk Statistics

sgw_voltesession_counter is added based on the *volteSession* flag. This counter indicates how many VoLTE sessions are present in the system at a particular moment.

If *volteSession* flag is true, counter gets incremented.

If *volteSession* flag isn't present (no VoLTE bearer is present in any PDN), the counter gets decremented.

Counter Name: sgw_voltesession_counter

Description: Current active VoLTE sessions present in the system.

Label:

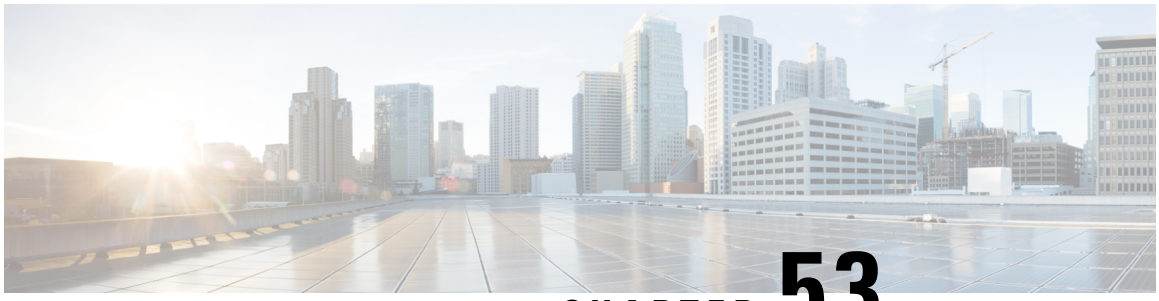
- LABEL_STAT: VolteSession

Sample Counter Output:

You can check the counter from the pod:

```
curl http://209.165.201.20:8080/metrics | grep "volte"

% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0#
HELP sgw_voltesession_counter Current Active Volte Session
# TYPE sgw_voltesession_counter gauge
sgw_voltesession_counter{app_name="smf",cluster="cn",data_center="cn",instance_id="0",
service_name="sgw-service",state="VolteSession"} 1
100  246k    0  246k    0     0  16.0M      0 --:--:-- --:--:-- --:--:-- 17.1M
root@sgw-service-n0-0:/opt/workspace#
```



CHAPTER 53

cnSGW-C Troubleshooting

- Description, on page 741
- Using CLI Data, on page 741
- Logs, on page 748

Description

This chapter provides information on using the command line interface (CLI) commands and logs for troubleshooting any issues that may arise during system operation.

Using CLI Data

This section describes the show and clear commands and the monitor commands that are used for troubleshooting

show subscriber and cdl show Commands

Table 264: Feature History

Feature Name	Release Information	Description
Enhanced session management and clearing options	2026.02.0	

Feature Name	Release Information	Description
		This feature introduces new filtering parameters to the existing show and clear subscriber commands. These enhancements allow operators to target sessions based on their activity state (idle or active) or the specific duration the session has been connected. Additionally, a max-count limit can be applied to restrict the number of sessions processed or displayed, providing better control over system resources during bulk operations. **Commands introduced:** The following options are added to the show subscriber, show subscriber count, and clear subscriber commands: • **sub-state**: Filters or clears sessions based on the subscriber's current activity state. • **active**: Targets sessions currently in an active state. • **idle**: Targets sessions currently in an idle state. • **connected-duration**: Filters or clears sessions based on the length of time the session has been connected. • • **greater-than <seconds>**: Targets sessions connected longer than the specified value. • **lesser-than <seconds>**: Targets sessions connected for less than the specified value. • **max-count**: Limits the number of subscriber sessions

Feature Name	Release Information	Description
		returned or cleared. • **lesser-than <value>**: Specifies the maximum number of sessions to be impacted by the command.
Supporting IPv6 Only eNB Insertion through Show and Clear Subscriber CLI commands	2024.02.0	Before you add IPv6 only eNBs in a network, all UPFs in a mesh must be IPv6 enabled for successful handovers of IPv4 only eNB sessions to IPv6 only eNB sessions. In addition, all sessions must have V4V6 tunnel before inserting a V6 only eNB. To support this IPV6 only eNB insertion, cnSGW-c includes the following CLI commands: • The **show subscriber nf-service sgw data-tunnel** *data_tunnel_type* and **show subscriber count nf-service sgw data-tunnel** *data_tunnel_type* CLI commands • The **clear subscriber nf-service sgw data-tunnel** *data_tunnel_type* CLI command **Default Setting**: Not Applicable

This section describes troubleshooting information.

- To display the SGW subscriber information, use the following commands:

  show subscriber namespace sgw imsi *imsi_value*

  show subscriber nf-service sgw imsi *imsi_value*

  show subscriber count { all }

```
show subscriber namespace sgw imsi 123456789012348
subscriber-details
{
  "subResponses": [
    {
      "status": true,
      "genericInfo": {
        "imsi": "imsi-123456789012348",
        "mei": "imei-123456786666660",
```

```
  "msisdn": "msisdn-223310101010101",
  "accessType": "EUTRAN",
  "plmnId": {
    "mcc": "123",
    "mnc": "456"
  },
  "sgwProfileName": "sgw1",
  "unAuthenticatedImsi": "No"
},
"s11cInterfaceInfo": {
  "sgwTeid": "[0x12000147] 301990215",
  "sgwIPv4Address": "209.165.201.19",
  "mmeTeid": "[0x62b5] 25269",
  "mmeIPv4Address": "209.165.201.20"
},
"pdnInfoList": {
  "totalPdn": 1,
  "pdnInfo": [
    {
      "pdnId": "PDN-1",
      "apn": "intershat",
      "attachType": "Initial Attach",
      "sgwRelocState": "N/A",
      "operatorPolicyName": "N/A",
      "dnnProfileName": "N/A",
      "defaultEbi": 5,
      "pdnType": "IPv4",
      "allocatedIPv4": "209.165.201.26",
      "apnSelectionMode": "Subscribed",
      "ambrUplink": "10 Kbps",
      "ambrDownlink": "20 Kbps",
      "s5cInterfaceInfo": {
        "sgwTeid": "[0x52000147] 1375732039",
        "sgwIPv4Address": "209.165.201.19",
        "pgwTeid": "[0x339a] 13210",
        "pgwIPv4Address": "209.165.201.18"
      },
      "sxaInterfaceInfo": {
        "selectedUP": "209.165.201.20",
        "upEpKey": "209.165.201.20:209.165.201.19",
        "cpSeid": "[0x1200014752000147] 1297038098512740679",
        "upSeid": "[0x2712] 10002"
      },
      "bearerInfoList": {
        "totalBearer": 1,
        "bearerInfo": [
          {
            "bearerId": "Bearer-1",
            "state": "Connected",
            "ebi": 5,
            "isDefaultBearer": true,
            "qosInfo": {
              "qci": 6,
              "arp": 113
            },
            "s1uInterfaceInfo": {
              "sgwTeid": "[0x62b7] 25271",
              "sgwIPv4Address": "209.165.200.226",
              "eNodeBTeid": "[0x62b8] 25272",
              "eNodeBIPv4Address": "209.165.201.20"
            },
            "s5uInterfaceInfo": {
              "sgwTeid": "[0x62b6] 25270",
              "sgwIPv4Address": "209.165.201.1",
```

```
                    "pgwTeid": "[0x339b] 13211",
                    "pgwIPv4Address": "209.165.201.18"
                  },
                  "chargingId": 303174163
                }
              ]
            },
            "uli": {
              "mcc": "123",
              "mnc": "456",
              "tac": "0x92a",
              "eci": "0x12d687"
            },
            "uetimeZone": {
              "timeZone": "+0:15",
              "dayLightSavingTime": "+1 hour"
            },
            "plmnType": "VISITOR"
            "dualStackEnabled": true,
            "dataTunnelType": "IPV4"
            "ipSrcViolation": "N/A",
            "mmeS4SgsnId": {
              "IPv4": "10.10.10.1"
            }
          }
        ]
      }
    }
  ]
}
```

```
show subscriber count all
subscriber-details
{
   "sessionCount": 50
}
```

- To display the session summary information, use the following command:

  cdl show sessions summary

```
cdl show sessions summary
message params: {session-summary cli session {0 100 0 [] 0 0}}
session {
    primary-key imsi-146062234105885
    unique-key [ 16777218 ]
    map-id 1
    instance-id 1
    version 1
    create-time 2020-04-27 16:18:24.225646626 +0000 UTC
    last-updated-time 2020-04-27 16:18:24.87241245 +0000 UTC
    purge-on-eval false
    next-eval-time 2020-05-04 16:18:24 +0000 UTC
    data-size 406
}
```

- To display the subscriber output based on the data tunnel type, use the following command:

  show subscriber nf-service sgw data-tunnel *data_tunnel_type*

```
show subscriber nf-service sgw data-tunnel IPV4V6
subscriber-details
{
  "subResponses": [
    [
      "id-index:1:0:32768",
```

```
      "id-index-key:1:0:globalKey:32768",
      "id-value:16777344",
      "imsi:imsi-123456789012348",
      "msisdn:msisdn-223310101010101",
      "imei:imei-123456786666660",
      "upf:10.1.12.212",
      "upfEpKey:10.1.12.212:10.1.10.78",
      "subscribertype:non-volte",
      "s5s8Ipv4:10.1.15.100",
      "s11Ipv4:10.1.12.212",
      "data-tunnel:IPV4V6",
      "namespace:sgw",
      "nf-service:sgw"
    ]
  ]
}
```

- To display the subscriber count output based on data tunnel type, use the following command:

  show subscriber count nf-service sgw data-tunnel *data_tunnel_type*

```
show subscriber count nf-service sgw data-tunnel IPV4V6
subscriber-details
{
  "sessionCount": 1
}
```

- To clear subscriber information, use the following commands:

  clear subscriber all

  clear subscriber nf-service sgw all

```
clear subscriber all
result
ClearSubscriber Request submitted
```

```
clear subscriber nf-service sgw all
result
ClearSubscriber Request submitted
```

- To clear the subscriber information using a data tunnel type, use the following command:

  clear subscriber nf-service sgw data-tunnel *data_tunnel_type*

```
clear subscriber nf-service sgw data-tunnel IPV4V6
result
Clear subscriber request submitted successfully for GR Instance ID 1. Waiting Time is
9.351706 seconds
```

Enhanced session management and clearing options

New filtering parameters are introduced to the existing **show subscriber**, **show subscriber count**, and **clear subscriber** CLI commands to enhance session management and clearing options in cnSGW.

The following options are added to the **show subscriber**, **show subscriber count**, and **clear subscriber** commands.

- **sub-state**: Filters or clears sessions based on the subscriber's current activity state.
  - **active**: Targets sessions currently in an active state.
  - **idle**: Targets sessions currently in an idle state.

Example:

```
show subscriber namespace sgw gr-instance 1 sub-state idle
show subscriber namespace sgw gr-instance 1 sub-state active
clear subscriber namespace sgw gr-instance 1 sub-state idle
clear subscriber namespace sgw gr-instance 1 sub-state active
show subscriber count namespace sgw gr-instance 1 sub-state idle
show subscriber count namespace sgw gr-instance 1 sub-state active
```

- **connected-duration**: Filters or clears sessions based on the length of time the session has been connected.
  - **greater-than <seconds>**: Targets sessions connected longer than the specified value.
  - **lesser-than <seconds>**: Targets sessions connected for less than the specified value.

Example:

```
show subscriber namespace sgw gr-instance 1 connected-duration greater-than <seconds>
show subscriber namespace sgw gr-instance 1 connected-duration lesser-than <seconds>
clear subscriber namespace sgw gr-instance 1 connected-duration greater-than <seconds>
clear subscriber namespace sgw gr-instance 1 connected-duration lesser-than <seconds>
show subscriber count namespace sgw gr-instance 1 connected-duration greater-than
<seconds>
show subscriber count namespace sgw gr-instance 1 connected-duration lesser-than <seconds>
```

- **max-count**: Limits the number of subscriber sessions returned or cleared.
  - **lesser-than <value>**: Specifies the maximum number of sessions to be impacted by the command.

Example:

```
show subscriber namespace sgw max-count { lesser-than <count> }
clear subscriber namespace sgw max-count { lesser-than <count> }
```

Logs

The system logging feature provides a common way to log the log messages across applications. Each log consists of the following components:

- Timestamp—Shows the date and time of the log creation.
- Log message—Shows the message of a specific log.
- Log level—Shows the level of importance of log message.
- Log tag—Shows the details of module name, component name, and interface name. A log tag is pre-created and passes during logging.

Logs for Event Failures

Table 265: Feature History

Feature Name	Release Information	Description
Event Failure Logs for Service Pods	2025.03.0	With this feature, the consistent event failure logs are enhanced to support the IDFT, LI, Suspend Notification, Resume Notification, and PDN Disconnect (GTPU Path Failure) procedures for the service pods.
Event Failure Logs for Service Pods	2025.01.0	With this feature, the consistent event failure logs are enhanced to support the Delete Bearer Command, Change Notification, and PDN Disconnect procedures for the service pods.
Event Failure Logs for Service Pods	2024.04.0	With this feature, the consistent event failure logs are enhanced to support the Create Bearer, Update Bearer, Delete Bearer, PDN Modify List, and Modify Bearer Command procedures for the service pods.

Feature Name	Release Information	Description
Event Failure Logs	2024.03.0	cnSGWc provides the following support: • Consistent event failure logs for PDN Setup, Idle or Active, PDN Modify, and PDN Disconnect procedures across pods • Configurable logs at pod type • Inclusion of request and response details in a single-line format The significant volume of unnecessary system-generated logs resulted in increased memory consumption, performance impact, and ineffective management and utilization of logs. To prevent these issues, the consistent error log message format across various pods is introduced for reduced memory consumption, minimized number of log generations by the system, and efficient troubleshooting. The single-line log format display enhances the readability. The significant volume of unnecessary system-generated logs resulted in increased memory consumption, performance impact, and ineffective management and utilization of logs. To prevent these issues, the consistent error log message format across various pods is introduced. The enhanced error logging for cnSGWc procedures provides significant improvements, such as reduced memory consumption, minimized number of log generations by the system, detailed, consistent, and configurable logging that help in effective debugging and system monitoring. **Default Setting:** Not Applicable

The error logging capabilities for cnSGWc procedures are enhanced for providing detailed, consistent, and configurable logging. These enhancements help in effective debugging and system monitoring. These enhancements are:

- Consistent log format—The single-line log format is standardized across different pods, such as REST endpoint and service pods, to ensure uniformity in how logs are recorded and interpreted.
- Enabling and disabling logging—An option to enable or disable logging is available at specific pod types. This option provides flexibility in managing log storage.
- Detailed log content—Logs include comprehensive details, such as primary key, interface, procedure details, message requests, and responses. This level of detail helps in thorough debugging and analysis.
- Log level management—By default, logs are written at the INFO level. You can enable the logs, as required. The log enablement helps in controlling the amount of log data generated and stored.

  By default, the log level is set to WARN, which ensures that logging is disabled by default. You can enable the logging, as required.
- Common logging interface—A common interface is implemented for event logging. All components use this interface for the same logging standards and formats.
- Log tags—Log tags are enhanced to allow you to enable or disable logs for specific pod types or services and provides granular control over logging.
- Supported logs—Logging is supported for various procedures, pods, and interfaces.

How it Works

To have the consistent log format across each pod, each component uses a common interface for event logging. A log uses the JSON format so that all the data appears in a single line. Logs are written at INFO level so that this level can be disabled by default and enabled, as required.

A log tag has the following format:

transaction.event.<pod-type>, where **<pod-type>** is the service name that a pod uses. For example, rest-ep and sgw-service.

You can enable or disable logs for a specific pod type or service using log tags.

The logs are written when the corresponding log level is set to INFO, DEBUG, or TRACE. However, the message request or response fields are populated only when the log level is set to DEBUG or TRACE.

Sample Log when Debug level is enabled

```
sgw-service-0 [INFO] [Transaction.go:1595] [transaction.event.sgw-service]
 {"TxnId":16,"StartTime":"2024-05-10T15:01:00+05:30","GRInstanceId":1,
"TxnType":"S5CreateSessReq","Priority":33,"SessionNamespace":"sgw(2)",
"CdlSliceName":"1","SubscriberId":"imsi-430967582185910","SessionPrimaryKey":
"imsi-430967582185910","SessionKeys":"imsi-430967582185910 (pk)
subscribertype:wps (nuk) 16777217 (uk) id-index:1:0:32768 (nuk)
id-index-key:1:0:globalKey:32768 (nuk) id-value:16777217 (nuk) imsi:
imsi-430967582185910 (nuk) msisdn:msisdn-9326737733 (nuk) imei:imeisv-1122334455667788
 (nuk) upf:192.168.56.20 (nuk) upfEpKey:192.168.56.20:192.168.56.10
(nuk) s11Ipv4:192.168.56.20 (nuk) s5s8Ipv4:192.168.56.30 (nuk)",
"SessionState":"Create_Session","ErrorMessage":"{\"ErrType\":3,\"ErrCause\":
{\"Value\":89,\"Pce\":false,\"Bce\":false,\"OrigInd\":false,\"OffendingIe\":
{\"Valid\":false,\"Tag\":0,\"Instance\":0,\"Value\":\"\"}},\"BrCtxtCause\":true,
\"SubfailReason\":89,\"SubfailReasonDetailed\":0,\"SubfailStr\":\
"S5 Create Session Response Failure\",\"SubfailReasonStr\":\"IPv4:192.168.56.30
```

```
IPv6:\",\"SourceDetails\":\"/opt/workspace/sgw_service/src/sgw-service/procedures
/pdnsetup/idlestate.go:963\"}","MessageRequest":{"Version":2,"TEIDflag":true,
"MsgPriority":true,"MsgTypeId":32,"MsgPriorityValue":10,"meta_data":{"from_ip"
:3232249867,"to_ip":3232249886,"to_port":2123,"intfType":3,"s5edscp":{"Value":12,
"Valid":true},"peerType":1},"MsgType":{"Create_Session_Request":{"IMSI":
"430967582185910","APN":"starent.com","AMBR":{"UL":119,"DL":135},"MEI":
"1122334455667788","MSISDN":"9326737733","Indication":{},"PAA":{"PDN_Type":1,
"IPv4":"0.0.0.0"},"RAT_Type":{"value":6},"Serving_Network":{"MCC":"123","MNC":
"765"},"ULI":{"uli_tai":{"mcc":"214","mnc":"365","val":4660}},"FQ_TEID":{"sgwCntrl":
{"IFace":6,"TEID":1358954497,"IPv4":"192.168.56.11"}},"Bearer_Context_List":
{"num_bearer_ctxt":1,"pbBearerCxt":[{"linked_ebi":{"value":5},"fqteid":{"sgwData":
{"TEID":1073807935,"IPv4":"192.168.56.60","IPv6":"4123::192:168:56:60"}},
"bearerQos":{"PL":2,"QCI":5,"arp":8}}]},"Charging_Characteristics":{"value":
"NBIAAA=="},"PDN_Type":{"value":1},"APN_Restriction":{},"Selection_Mode":
{"value":1}}}},"MessageResponse":{"Version":2,"TEIDflag":true,"MsgPriority":
true,"MsgLength":19,"TIED":1358954497,"Seq":1,"MsgTypeId":33,"MsgPriorityValue":10,
"meta_data":{"from_ip":3232249886,"to_ip":3232249867,"intfType":3},"MsgType":
{"Create_Session_Response":{"Cause":{"Cause_Value":89},"Recovery":{"value":10}}}}}
```


Sample Log when Info level is enabled


```
sgw-service-0 [INFO] [Transaction.go:1607] [transaction.event.sgw-service]
 {"TxnId":7,"StartTime":"2024-06-20T12:10:05+05:30","GRInstanceId":1,"TxnType":
"S5CreateSessReq","Priority":33,"SessionNamespace":"sgw(2)","CdlSliceName":"1",
"SubscriberId":"imsi-430967582185910","SessionPrimaryKey":"imsi-430967582185910",
"SessionKeys":"imsi-430967582185910 (pk) subscribertype:wps (nuk) 16777217 (uk)
id-index:1:0:32768 (nuk) id-index-key:1:0:globalKey:32768 (nuk) id-value:16777217
(nuk) imsi:imsi-430967582185910 (nuk) msisdn:msisdn-9326737733 (nuk) imei:
imeisv-1122334455667788 (nuk) upf:192.168.56.20 (nuk)
upfEpKey:192.168.56.20:192.168.56.10 (nuk) s11Ipv4:192.168.56.20 (nuk)
s5s8Ipv4:192.168.56.30 (nuk)","SessionState":"Create_Session","ErrorMessage":
"{\"ErrType\":3,\"ErrCause\":{\"Value\":89,\"Pce\":false,\"Bce\":false,\"OrigInd\"
:false,\"OffendingIe\":{\"Valid\":false,\"Tag\":0,\"Instance\":0,\"Value\":\"\"}},
\"BrCtxtCause\":true,\"SubfailReason\":89,\"SubfailReasonDetailed\":0,\"SubfailStr\":
\"S5 Create Session Response Failure\",\"SubfailReasonStr\":\"IPv4:192.168.56.30
IPv6:\",\"SourceDetails\":\"/opt/workspace/sgw_service/src/sgw-service/procedures
/pdnsetup/idlestate.go:971\"}"}
```


Supported Logs

This feature supports logs for cnSGWc to provide detailed error information across various procedures, pods, and interfaces. Following table lists the supported logs.

Table 266: Supported Logs

cnSGWc Procedures with Supported Logging	Pod Involved in Logging	cnSGWc Interfaces Associated with Pod and Procedure
PDN Setup	SERVICE	S11, S5, SXA, RMGR
Idle/Active	SERVICE	S11, S5, SXA
PDN Modify	SERVICE	S11, SXA, S5
PDN Disconnect (DSR/DBR)	SERVICE	S11, S5, SXA, RMGR
Create Bearer	SERVICE	S11, S5, SXA
Update Bearer	SERVICE	S11, S5, SXA
Delete Bearer (Dedicated)	SERVICE	S11, S5, SXA

cnSGWc Procedures with Supported Logging	Pod Involved in Logging	cnSGWc Interfaces Associated with Pod and Procedure
PDN Modify List	SERVICE	S11, S5, SXA
Modify Bearer Command	SERVICE	S11, S5, SXA
Delete Bearer Command	SERVICE	S11, S5, SXA
Change Notification	SERVICE	S11, S5, SXA
PDN Disconnect	SERVICE	S11, S5, SXA, RMGR
IDFT	SERVICE	S11, SXA
LI	SERVICE	X2, X3
Suspend Notification	SERVICE	S11, S5, SXA
Resume Notification	SERVICE	S11, S5, SXA
PDN Disconnect (GTPU Path Failure)	SERVICE	S11, S5, SXA, RMGR

Enable Event Logging

Use the following procedure to enable or disable the event logs of the cnSGWc service. The appropriate log level configuration using the CLI command allows you to control the amount and type of log data generated. Hence, this configuration helps in effective monitoring, troubleshooting, and performance management.

Procedure

Step 1 Specify the log level for event logging.

logging name transaction.event.<*podname*> **level application** *application_log_level*

The <pod-type> is the service name that a pod uses.

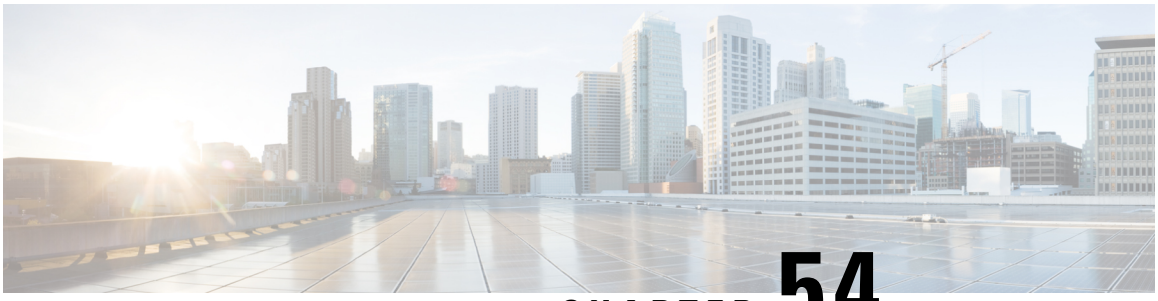
Example:

```
 logging name transaction.event.sgw-service level application [ debug | error | info | off | trace | warn ]
```

Step 2 [Optional] Use **show running-config logging** command to verify the configured application event logging level.

Example:

```
show running-config logging
   logging level tracing debug
   logging name infra.config.core level application trace
   logging name infra.config.core level transaction trace
   logging name infra.config.core level tracing off
   logging name infra.message_log.core level transaction trace
   logging name transaction.event.sgw-service level application debug
```



CHAPTER 54

Sample cnSGW-C Configuration

- Sample Configuration, on page 755

Sample Configuration

The following is a sample configuration.

```
show running-config
profile compliance comp1
 service nsmf-pdusession
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service namf-comm
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service n1
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service n2
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service nudm-sdm
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service nudm-uecm
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service nnrf-disc
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service nnrf-nfm
  version uri v1
```

```
  version full 1.0.0
  version spec 15.4.0
 exit
 service npcf-smpolicycontrol
  version uri v1
  version full 1.0.0
  version spec 15.4.0
 exit
 service nchf-convergedcharging
  version uri v1
  version full 1.0.0
  version spec 15.3.0
 exit
exit
profile network-element amf amf1
 nf-client-profile        AP1
 failure-handling-profile FH3
 query-params [ dnn ]
exit
profile network-element udm udm1
 nf-client-profile        UP1
 failure-handling-profile FH4
 query-params [ dnn ]
exit
profile network-element pcf pcf1
 nf-client-profile        PP1
 failure-handling-profile FH1
 query-params [ dnn ]
 rulebase-prefix          cbn#
 predefined-rule-prefix   crn#
exit
profile network-element chf chf1
 nf-client-profile         CP1
 failure-handling-profile  FH2
 query-params [ dnn ]
 nf-client-profile-offline CP2
exit
profile network-element chf chgser1
exit
profile network-element upf upf1
 node-id      upf1@sgw.com
 n4-peer-address ipv4 209.165.200.234
 n4-peer-port 8805
 dnn-list     [ cisco.com intershat starent.com ]
 capacity     65535
 priority     65535
exit
profile upf-group group1
 failure-profile FH1
exit
profile icmpv6 icmpprf1
 options virtual-mac b6:6d:57:45:45:45
exit
profile charging chgprf1
 method [ offline ]
exit
profile charging-characteristics 1
 charging-profile chgprf1
exit
profile failure-handling FH1
 interface pfcp
  message N4SessionEstablishmentReq
   cause-code pfcp-entity-in-congestion action retry-terminate max-retry 2
   cause-code system-failure action terminate
```

```
    cause-code service-not-supported action terminate
    cause-code no-resource-available action retry-terminate max-retry 3
    cause-code no-response-received action retry-terminate max-retry 1
    cause-code reject action terminate
   exit
   message N4SessionModificationReq
    cause-code mandatory-ie-incorrect action terminate
    cause-code session-ctx-not-found action terminate
    cause-code reject action terminate
   exit
  exit
 exit
exit
profile failure-handling gtp1
 interface gtpc message S5S8CreateBearerReq
  cause-code temp-fail
   action retry timeout 1000 max-retry 2
  exit
 exit
 interface gtpc message S5S8UpdateBearerReq
  cause-code temp-fail
   action retry timeout 1000 max-retry 2
  exit
 exit
 interface gtpc message S5S8DeleteBearerReq
  cause-code temp-fail
   action retry timeout 1000 max-retry 2
  exit
 exit
exit
profile access access1
 n26 idft enable timeout 15
 n2 idft enable timeout 15
 gtpc gtpc-failure-profile gtp1
exit
profile dnn default-profile
 network-element-profiles chf chf1
 network-element-profiles amf amf1
 network-element-profiles pcf pcf1
 network-element-profiles udm udm1
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:47
 ssc-mode 2 allowed [ 3 ]
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn starent.com
exit
profile dnn intershat
 network-element-profiles chf chf1
 network-element-profiles amf amf1
 network-element-profiles pcf pcf1
 network-element-profiles udm udm1
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:47
 ssc-mode 2 allowed [ 3 ]
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn intershat
 dcnr             true
exit
profile dnn intershat1
 network-element-profiles chf chf1
 network-element-profiles amf amf1
 network-element-profiles pcf pcf1
 network-element-profiles udm udm1
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:48
```

```
 pcscf-profile    PCSCF_Prof_2
 ssc-mode 1
 session type IPV4
exit
profile dnn intershat2
 network-element-profiles chf chf
 network-element-profiles amf amf
 network-element-profiles pcf pcf
 network-element-profiles udm udm
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:49
 ssc-mode 2 allowed [ 3 ]
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn intershat2
exit
profile dnn starent.com
 network-element-profiles chf chf1
 network-element-profiles amf amf1
 network-element-profiles pcf pcf1
 network-element-profiles udm udm1
 charging-profile chgprf1
 virtual-mac      b6:6d:47:47:47:47
 ssc-mode 2 allowed [ 3 ]
 session type IPV4 allowed [ IPV6 IPV4V6 ]
 upf apn starent.com
exit
profile qos abc
 ambr ul "250 Kbps"
 ambr dl "500 Kbps"
 qi5      7
 arp priority-level 14
 arp preempt-cap NOT_PREEMPT
 arp preempt-vuln PREEMPTABLE
 priority 120
 max data-burst 2000
exit
profile nf-client nf-type udm
 udm-profile UP1
  locality LOC1
   priority 30
   service name type nudm-sdm
    endpoint-profile EP1
     capacity   30
     uri-scheme http
     version
      uri-version v2
      exit
     exit
     endpoint-name EP1
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8001
     exit
    exit
   exit
   service name type nudm-uecm
    endpoint-profile EP1
     capacity   30
     uri-scheme http
     endpoint-name EP1
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8001
     exit
    exit
   exit
```

```
   service name type nudm-ee
    endpoint-profile EP1
     capacity       30
     api-uri-prefix PREFIX
     api-root       ROOT
     uri-scheme     http
     endpoint-name EP1
      priority 56
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8001
     exit
    exit
   exit
  exit
 exit
exit
profile nf-client nf-type pcf
 pcf-profile PP1
  locality LOC1
   priority 30
   service name type npcf-am-policy-control
    endpoint-profile EP1
     capacity   30
     uri-scheme http
     endpoint-name EP1
      priority 56
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8003
     exit
    exit
   exit
   service name type npcf-smpolicycontrol
    endpoint-profile EP1
     capacity   30
     uri-scheme http
     endpoint-name EP1
      priority 56
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8003
     exit
    exit
   exit
  exit
 exit
exit
profile nf-client nf-type amf
 amf-profile AP1
  locality LOC1
   priority 30
   service name type namf-comm
    endpoint-profile EP2
     capacity   30
     uri-scheme http
     endpoint-name EP1
      priority 56
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8002
     exit
    exit
   exit
  exit
 exit
exit
profile nf-client nf-type chf
```

```
 chf-profile CP1
  locality LOC1
   priority 30
   service name type nchf-convergedcharging
    endpoint-profile EP1
     capacity   30
     uri-scheme http
     version
      uri-version v2
      exit
     exit
     endpoint-name EP1
      priority 56
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 8004
     exit
    exit
   exit
  exit
 exit
 chf-profile CP2
  locality LOC1
   priority 31
   service name type nchf-convergedcharging
    endpoint-profile EP1
     capacity   30
     uri-scheme http
     version
      uri-version v2
      exit
     exit
     endpoint-name EP1
      priority 56
      primary ip-address ipv4 209.165.201.21
      primary ip-address port 9040
     exit
    exit
   exit
  exit
 exit
exit
profile nf-pair nf-type UDM
 nrf-discovery-group udmdiscovery
 locality client  LOC1
 locality preferred-server LOC1
 locality geo-server GEO
exit
profile nf-pair nf-type AMF
 nrf-discovery-group udmdiscovery
 locality client  LOC1
 locality preferred-server LOC1
 locality geo-server GEO
exit
profile nf-pair nf-type PCF
 nrf-discovery-group udmdiscovery
 locality client  LOC1
 locality preferred-server LOC1
 locality geo-server GEO
exit
profile nf-pair nf-type UPF
 nrf-discovery-group udmdiscovery
 locality client  LOC1
 locality preferred-server LOC1
 locality geo-server GEO
```

```
exit
profile nf-pair nf-type CHF
 nrf-discovery-group udmdiscovery
 locality client  LOC1
 locality preferred-server LOC1
 locality geo-server GEO
exit
profile nf-client-failure nf-type udm
 profile failure-handling FH4
  service name type nudm-sdm
   message type UdmSdmGetUESMSubscriptionData
    status-code httpv2 403
     retry  3
     action retry-and-ignore
    exit
    status-code httpv2 404
     action continue
    exit
    status-code httpv2 413
     retry  3
     action retry-and-continue
    exit
    status-code httpv2 501,504
     retry  3
     action retry-and-terminate
    exit
    status-code httpv2 503
     action terminate
    exit
   exit
   message type UdmSdmSubscribeToNotification
    status-code httpv2 403
     retry  3
     action retry-and-ignore
    exit
    status-code httpv2 404
     action continue
    exit
    status-code httpv2 413
     retry  3
     action retry-and-continue
    exit
    status-code httpv2 501,504
     retry  3
     action retry-and-terminate
    exit
    status-code httpv2 503
     action terminate
    exit
   exit
  exit
  service name type nudm-uecm
   message type UdmUecmRegisterSMF
    status-code httpv2 403
     retry  3
     action retry-and-ignore
    exit
    status-code httpv2 404
     action continue
    exit
    status-code httpv2 413
     retry  3
     action retry-and-continue
    exit
```

```
    status-code httpv2 501,504
     retry  3
     action retry-and-terminate
    exit
    status-code httpv2 503
     action terminate
    exit
   exit
  exit
 exit
exit
profile nf-client-failure nf-type pcf
 profile failure-handling FH1
  service name type npcf-smpolicycontrol
   message type PcfSmpolicycontrolCreate
    status-code httpv2 0,403
     action retry-and-ignore
    exit
    status-code httpv2 400
     action continue
    exit
    status-code httpv2 404
     action terminate
    exit
    status-code httpv2 500
     retry  2
     action retry-and-ignore
    exit
    status-code httpv2 503
     retry  2
     action retry-and-continue
    exit
   exit
   message type PcfSmpolicycontrolUpdate
    status-code httpv2 0,403
     action retry-and-ignore
    exit
    status-code httpv2 400
     action continue
    exit
    status-code httpv2 404
     action terminate
    exit
    status-code httpv2 500
     retry  2
     action retry-and-ignore
    exit
    status-code httpv2 503
     retry  2
     action retry-and-continue
    exit
   exit
   message type PcfSmpolicycontrolDelete
    status-code httpv2 0,403
     action retry-and-ignore
    exit
    status-code httpv2 400
     action continue
    exit
    status-code httpv2 404
     action terminate
    exit
    status-code httpv2 500
     retry  2
```

```
     action retry-and-ignore
    exit
    status-code httpv2 503
     retry  2
     action retry-and-continue
    exit
   exit
  exit
 exit
exit
profile nf-client-failure nf-type chf
 profile failure-handling FH2
  service name type nchf-convergedcharging
   message type ChfConvergedchargingCreate
    status-code httpv2 0,500,504
     action continue
    exit
    status-code httpv2 400,404
     retry  3
     action retry-and-terminate
    exit
    status-code httpv2 403
     retry  3
     action retry-and-ignore
    exit
    status-code httpv2 503
     action terminate
    exit
   exit
   message type ChfConvergedchargingUpdate
    status-code httpv2 0,500,504
     action continue
    exit
    status-code httpv2 400,404
     retry  3
     action retry-and-terminate
    exit
    status-code httpv2 403
     retry  3
     action retry-and-ignore
    exit
    status-code httpv2 503
     action terminate
    exit
   exit
   message type ChfConvergedchargingDelete
    status-code httpv2 0,500,504
     action continue
    exit
    status-code httpv2 400,404
     retry  3
     action retry-and-terminate
    exit
    status-code httpv2 403
     retry  3
     action retry-and-ignore
    exit
    status-code httpv2 503
     action terminate
    exit
   exit
  exit
 exit
exit
```

```
profile smf smf1
 locality      LOC1
 allowed-nssai [ slice1 ]
 plmn-id mcc 123
 plmn-id mnc 456
 service name nsmf-pdu
  type               pdu-session
  schema             http
  service-id         1
  version            1.Rn.0.0
  http-endpoint base-url http://smf-service
  icmpv6-profile     icmpprf1
  compliance-profile comp1
  access-profile     access1
  subscriber-policy  polSub
 exit
exit
profile sgw sgw1
 sgw-charging-threshold thre1
 sgw-charging-profile   ch1
 locality               LOC2
 fqdn                   cisco.com.apn.epc.mnc456.mcc123
 charging-mode          gtpp
exit
profile sgw-charging-threshold thre1
 cc profile value 1
  volume total 100000
  buckets  1
  duration 60
 exit
 cc profile value 2
  volume uplink 100000
  volume downlink 100000
  buckets  1
  duration 120
 exit
exit
profile sgw-charging-profile ch1
 gtpp-triggers volume-limit enable
 gtpp-triggers time-limit enable
 gtpp-profile pf1
exit
profile gtpp-profile pf1 gtpp
 local-storage
  file
   rotation
    volume        5
    cdr-count     1000
    time-interval 60
   exit
   name
    prefix               NYPCF508
    format               .%Y-%m-%d%H-%M-%S.%4Q
    max-file-seq-num     4
    start-file-seq-num   1
    recover-file-seq-num false
   exit
   purge-processed-files purge-interval 10
   format custom5
  exit
  push
   encrypted-url
"$8$6vhjkoHt8RL2noFs/ON6ZJavTDzwGS2KUn/YqlBzgkezFmx5SzvnARYzAdVacCSyCirYOvcC\nTFnHpBNim3QY3Q=="
```

```
  exit
 exit
 dictionary custom24
exit
policy subscriber polSub
 precedence 1
  sst              02
  sdt              Abf123
  serving-plmn mcc 123
  serving-plmn mnc 456
  supi-start-range 100000000000001
  supi-stop-range  999999999999999
  gpsi-start-range 1000000000
  gpsi-stop-range  9999999999
  operator-policy  opPol1
 exit
 precedence 511
  operator-policy defOprPol1
 exit
exit
policy operator defOprPol1
 policy dnn         defPolDnn
 policy network-capability nc1
exit
policy operator opPol1
 policy dnn         polDnn
 policy network-capability nc1
exit
policy dnn defPolDnn
 profile default-profile
 dnn dnn2 profile profile2
 dnn intershat profile intershat
 dnn intershat1 profile intershat1
 dnn starent.com profile starent.com
exit
policy dnn polDnn
 profile default-profile
 dnn dnn2 profile profile2
 dnn intershat profile intershat
 dnn intershat1 profile intershat1
 dnn intershat2 profile intershat2
 dnn starent.com profile starent.com
exit
policy network-capability nc1
 nw-support-local-address-tft true
exit
nssai name slice1
 sst 2
 sdt Abf123
 dnn [ dnn1 intershat intershat1 intershat2 ]
exit
ipam
 instance 1
  source local
  address-pool poolv4
   vrf-name ISP
   tags
    dnn starent.com
   exit
   ipv4
    split-size
     per-cache 1024
     per-dp    256
    exit
```

```
  address-range 209.165.202.129 209.165.200.253
 exit
exit
address-pool poolv4DNN2
 vrf-name ISP
 tags
  dnn intershat1
 exit
 ipv4
  split-size
   per-cache 1024
   per-dp    256
  exit
  address-range 209.165.200.241 209.165.200.244
 exit
exit
address-pool poolv4DNN3
 vrf-name ISP
 static
 tags
  dnn intershat2
 exit
 ipv4
  split-size
   per-cache 512
   per-dp    512
  exit
  address-range 209.165.200.247 209.165.200.248
 exit
 ipv6
  prefix-ranges
   split-size
    per-cache 8192
    per-dp    8192
   exit
   prefix-range 2002:db0:: length 48
  exit
 exit
exit
address-pool poolv4vDNN
 vrf-name ISP
 tags
  dnn intershat1
 exit
 ipv4
  split-size
   per-cache 1024
   per-dp    256
  exit
  address-range 209.165.200.245 209.165.200.244
 exit
exit
address-pool poolv6
 vrf-name ISP
 tags
  dnn intershat
 exit
 ipv6
  prefix-ranges
   split-size
    per-cache 8192
    per-dp    1024
   exit
   prefix-range 2001:db0:: length 48
```

```
    exit
   exit
  exit
  address-pool poolv6DNN2
   vrf-name ISP
   tags
    dnn intershat1
   exit
   ipv6
    prefix-ranges
     split-size
      per-cache 8192
      per-dp    1024
     exit
     prefix-range 2001:ef0:: length 48
    exit
   exit
  exit
  address-pool poolv6vDNN
   vrf-name ISP
   tags
    dnn intershat1
   exit
   ipv6
    prefix-ranges
     split-size
      per-cache 8192
      per-dp    1024
     exit
     prefix-range 2001:ab0:: length 48
    exit
   exit
  exit
 exit
exit
cdl system-id          1
cdl enable-geo-replication true
cdl deployment-model   small
cdl zookeeper replica 1
cdl remote-site 2
 db-endpoint host 209.165.202.157
 db-endpoint port 8882
 kafka-server 209.165.202.157 10001
 exit
exit
cdl datastore session
 geo-remote-site [ 2 ]
 slice-names     [ cnSGW1 cnSGW2 ]
 endpoint replica 1
 endpoint external-ip 209.165.202.156
 endpoint external-port 8882
 index map    1
 index write-factor 1
 slot replica 1
 slot map     1
 slot write-factor 1
 features instance-aware-notification enable true
 features instance-aware-notification system-id 1
  slice-names [ cnSGW1 ]
 exit
 features instance-aware-notification system-id 2
  slice-names [ cnSGW2 ]
 exit
exit
```

```
cdl kafka replica 1
cdl kafka external-ip 209.165.202.156 10001
exit
etcd replicas 1
instance instance-id 1
 endpoint li
  replicas 1
  vip-ip 209.165.200.237
 exit
 endpoint nodemgr
  replicas 1
  nodes    1
 exit
 endpoint gtp
  replicas 1
  interface s5
   vip-ip 209.165.201.11
  exit
  interface s5e
   vip-ip 209.165.201.21
  exit
  interface s11
   vip-ip 209.165.200.237
  exit
 exit
 endpoint pfcp
  replicas 1
  interface sxa
   heartbeat
    interval 0
   exit
  exit
  interface n4
   heartbeat
    interval               0
    retransmission-timeout 3
    max-retransmissions    5
   exit
  exit
 exit
 endpoint radius-dns
  replicas 1
  vip-ip 209.165.201.21
 exit
 endpoint service
  replicas 1
 exit
 endpoint protocol
  replicas     1
  internal-vip 209.165.201.11
  vip-ip 209.165.201.21
  interface sxa
   vip-ip 209.165.201.21
  exit
  interface n4
   vip-ip 209.165.201.11
  exit
 exit
 endpoint gtpprime
  replicas 2
  nodes    1
 exit
 endpoint sgw-service
  replicas 1
```

```
 exit
 endpoint geo
  replicas 1
  nodes    2
  interface geo-internal
   vip-ip 209.165.200.233 vip-port 7001
  exit
  interface geo-external
   vip-ip 209.165.200.234 vip-port 7002
  exit
 exit
 endpoint sbi
  replicas 1
  vip-ip 209.165.201.21
 exit
 endpoint bgpspeaker
  replicas 1
  nodes    2
 exit
exit
instance instance-id 2
 endpoint li
  replicas 1
  vip-ip 209.165.200.238
 exit
 endpoint nodemgr
  replicas 1
  nodes    1
 exit
 endpoint gtp
  replicas 1
  interface s5
   vip-ip 209.165.201.12
  exit
  interface s5e
   vip-ip 209.165.201.141
  exit
  interface s11
   vip-ip 209.165.200.238
  exit
 exit
 endpoint pfcp
  replicas 1
  interface sxa
   heartbeat
    interval 0
   exit
  exit
  interface n4
   heartbeat
    interval               0
    retransmission-timeout 3
    max-retransmissions    5
   exit
  exit
 exit
 endpoint radius-dns
  replicas 1
  vip-ip 209.165.201.141
 exit
 endpoint service
  replicas 1
 exit
 endpoint protocol
```

```
  replicas     1
  internal-vip 209.165.201.11
  vip-ip 209.165.201.141
  interface sxa
   vip-ip 209.165.201.141
  exit
  interface n4
   vip-ip 209.165.201.12
  exit
 exit
 endpoint gtpprime
  replicas 2
  nodes    1
 exit
 endpoint sgw-service
  replicas 1
 exit
 endpoint geo
  replicas 1
  nodes    2
  interface geo-internal
   vip-ip 209.165.200.235 vip-port 7001
  exit
  interface geo-external
   vip-ip 209.165.200.236 vip-port 7002
  exit
 exit
 endpoint sbi
  replicas 1
  vip-ip 209.165.201.141
 exit
 endpoint bgpspeaker
  replicas 1
  nodes    2
 exit
exit
logging level application debug
logging level transaction debug
logging level tracing debug
logging name gtpp-ep0.application.config level application debug
logging name gtpp-ep0.application.gen level application trace
logging name gtpp-ep1.application.config level application debug
logging name gtpp-ep1.application.gen level application trace
logging name infra.cdr.core level application debug
logging name infra.cdr_sftp.core level application debug
logging name infra.config.core level application trace
logging name infra.config.core level transaction trace
logging name infra.config.core level tracing off
logging name infra.message_log.core level transaction trace
router bgp 65061
 bfd interval 250000 min_rx 250000 multiplier 3
 interface v4001
  neighbor 209.165.202.131 remote-as 65060 fail-over bfd
 exit
 policy-name allow-all ip-prefix 209.165.201.30/0 mask-range 0..32
exit
deployment
 app-name     smf
 cluster-name Local
 dc-name      DC
 model        small
exit
k8 label protocol-layer key disktype value ssd
exit
```

```
geomonitor podmonitor pods bgpspeaker-pod
 retryCount            1
 retryInterval         200
 retryFailOverInterval 200
 failedReplicaPercent  40
exit
geomonitor podmonitor pods gtpp-ep
 retryCount            1
 retryInterval         200
 retryFailOverInterval 200
 failedReplicaPercent  40
exit
geomonitor podmonitor pods li-ep
 retryCount            1
 retryInterval         200
 retryFailOverInterval 200
 failedReplicaPercent  40
exit
geomonitor podmonitor pods sgw-service
 retryCount            1
 retryInterval         200
 retryFailOverInterval 200
 failedReplicaPercent  40
exit
instances instance 1
 system-id  DCNAME001
 cluster-id CLUSTER0001
 slice-name cnSGW1
exit
instances instance 2
 system-id  DCNAME002
 cluster-id CLUSTER0002
 slice-name cnSGW2
exit
local-instance instance 1
system mode running
helm default-repository cn
helm repository cn
 access-token
sgw-deployer.gen:AKCp8ihVrCfvm9puwTSt8oKGK6HxP1Fn8sLY5fzqWyR3NhrBmjjJrUHaxfZD3ziQpiLkAy1Q3
 url
https://engci-maven-master.cisco.com/artifactory/smi-fuse-internal-snapshot/mobile-cnat-cn/cn-products/rel-2021.02/
exit
k8s name           cn
k8s namespace      cn
k8s nf-name        smf
k8s registry       dockerhub.cisco.com/smi-fuse-docker-internal
k8s single-node    true
k8s use-volume-claims true
k8s image-pull-secrets regcred
k8s ingress-host-name 209.165.200.235.nip.io
aaa authentication users user admin
 uid        117
 gid        117
 password   $1$g8J36yTY$1g/tM5a9pdsGMnKcspnxD.
 ssh_keydir /tmp/admin/.ssh
 homedir    /tmp/admin
exit
aaa ios level 0
 prompt "\h> "
exit
aaa ios level 15
 prompt "\h# "
exit
```

```
aaa ios privilege exec
 level 0
  command action
  exit
  command autowizard
  exit
  command enable
  exit
  command exit
  exit
  command help
  exit
  command startup
  exit
 exit
 level 15
  command configure
  exit
 exit
exit
nacm write-default deny
nacm groups group LI
 user-name [ liadmin ]
exit
nacm groups group LI2
 user-name [ liadmin2 ]
exit
nacm groups group LI3
 user-name [ liadmin3 ]
exit
nacm groups group admin
 user-name [ admin ]
exit
nacm rule-list admin
 group [ admin ]
 rule li-deny-tap
  module-name       lawful-intercept
  path              /lawful-intercept
  access-operations *
  action            deny
 exit
 rule li-deny-clear
  module-name       tailf-mobile-smf
  path              /clear/lawful-intercept
  access-operations *
  action            deny
 exit
 rule any-access
  action permit
 exit
exit
nacm rule-list confd-api-manager
 group [ confd-api-manager ]
 rule any-access
  action permit
 exit
exit
nacm rule-list ops-center-security
 group [ * ]
 rule change-self-password
  module-name       ops-center-security
  path              /smiuser/change-self-password
  access-operations exec
  action            permit
```

```
 exit
 rule smiuser
  module-name       ops-center-security
  path              /smiuser
  access-operations exec
  action            deny
 exit
exit
nacm rule-list lawful-intercept
 group [ LI LI2 LI3 ]
 rule li-accept-tap
  module-name       lawful-intercept
  path              /lawful-intercept
  access-operations *
  action            permit
 exit
 rule li-accept-clear
  module-name       tailf-mobile-smf
  path              /clear/lawful-intercept
  access-operations *
  action            permit
 exit
exit
nacm rule-list any-group
 group [ * ]
 rule li-deny-tap
  module-name       lawful-intercept
  path              /lawful-intercept
  access-operations *
  action            deny
 exit
 rule li-deny-clear
  module-name       tailf-mobile-smf
  path              /clear/lawful-intercept
  access-operations *
  action            deny
 exit
exit
```