SMF Serviceability

Feature Summary and Revision History

Summary Data

Table 1: Summary Data

Applicable Product(s) or Functional Area	SMF
Applicable Platform(s)	SMI
Feature Default Setting	Enabled – Always-on
Related Changes in this Release	Not Applicable
Related Documentation	Not Applicable

Revision History

Table 2: Revision History

Revision Details	Release
First introduced.	2023.01.0

Feature Description

The SMF logs and serviceability feature perform in the following modes:

- It helps and captures capabilities to enable limited debug logs in the production with a chain of troubleshooting tools.
- These further help in improving the time taken in finding a root cause of any issue found during the production.

This feature allows the operator to enable specific LogTag to debug call failures and procedure failures, to facilitate a better RCA in the production. It has the following characteristics:

- The SMF must provide a tool or utility to generate core dumps that help in downloading and investigating the problem further.
- The SMF must print the SUPI of the failed subscriber along with the error details in the logs.
- The required subscriber information MonSub helps in debugging a particular subscriber.

Relationships

The following modules are associated with this feature:

- Call Failure Logs, on page 7
- Procedure Failure Logs, on page 8
- Generic Procedure Failure Logs, on page 11
- Additional Call Flow Failure Logs, on page 12
- Event Trace Logs, on page 15
- Call Flow Statistics Logs, on page 19
- Core Dump Utility Logs, on page 20
- Monitor Subscriber (MonSub) Logs, on page 26
- N40 Additional Logs and Statistics, on page 30
- N7 Additional Logs and Statistics, on page 32

How it Works

This section describes how this feature works.

Log Instances

The log instance generates messages, used for identifying issues, deployment status, and performance tuning. The application infrastructure provides a common way to enable log messages across applications. Each log instance has the following:

- Timestamp
- Log message
- Log level
- LogTag

LogTag

The LogTag is used as a filter to enable or disable the specific type of log messages. Before or during the logging session, they are precreated, evaluated, and exempted. It consists of the following:

- Module name
- Component name
- Interface name

Creating a LogTag

The following is a sample example of creating a LogTag:

```
common.LogTagN7RestEp = appCtx.RegisterLogTag("rest_ep", "app", "n7")
```

Logging a Message

The following is a sample example of a LogTag message logging:

```
appCtx.Info(common.LogTagN7RestEp, "Starting rest-ep app")
```

Log Levels

The application infrastructure provides upto six variants of log levels, which can be used as filter along with LogTag to enable or disable specific logs. Each level represents the level of importance of a log message, which can be used while troubleshooting. The following table lists log levels and their usage:

Table 3: Log Levels

Log Levels	Usage
Error	Used when there's incorrectness leading to serious issues.
Warning	Used to notify and warn, when there's an occurrence of any of the following scenarios: • Something serious is about to happen. • If there's an activity which is erroneously running • Continuously giving error notes • Not attended scenarios
Info	Used for normal expected behavior such as starting an application or stopping the same, and so on.
Debug	Used to provide more information required to debug problems.
Trace	Used to provide extensive information. Also, used in monitoring routines, where the same debug log keeps coming periodically.
Off	Used when there isn't any meaning in the logging information. Configured in the CLI to turn off or turn on the logging activity.

Log Level Order

Every time, the application infrastructure logs any log message, it matches the log level, and the LogTag with the configured log setting, before logging.

Table 4: Log Levels

Log Levels (Order)	Usage
Error (0)	Matches error logs.
Warn (1)	Matches warn and error logs.
Info (2)	Matches info, warn, and error logs.
Debug (3)	Matches debug, info, warn, and error logs.
Trace (4)	Matches trace, debug, info, warn, and error logs.
Off (5)	Matches no log to disable errors.

SMF Logs

The SMF supports operators to enable specific LogTag. The following is a list of modules which support the implementation of this feature.

1. **Call Failure LogTag:**
    - Enabled in the CLI mode
    - The SMF must enable the following logs based on the disconnect reason:
        - Error
        - Warning
        - Debug
    - It prints the following log parameters:
        - Transaction ID
        - Disconnect reason
        - Event Trace
        - Session Keys
        - Error Category (For major disconnect reason)

2. **Procedure Failure LogTag**:
    - Enabled in the CLI mode
    - The SMF must enable the following logs based on the disconnect reason:
        - Warning
    - It prints the following log parameters:
        - Transaction ID
        - Procedure Name
        - Event Trace
        - Detailed Error: It includes the following:
            - Failure reason or cause
            - Disconnect reason (if applicable)
            - Failure metrics

3. **Additional Call Flow Failure LogTag**:
    - Enabled for specific reasons or conditions
    - It prints the following log parameters:
        - Transaction ID

- Event Trace
- Call flow-specific error messages

- The supported log levels:
  - Error
  - Warning
  - Info
  - Debug

4. **Additional Generic Procedure Failure LogTag**:
   - Enabled in the CLI mode
   - The SMF must not enable the Event Trace based on the disconnect reason.
   - If the operator needs failure logs for all procedures, then the operator can enable this LogTag.

Note The operator needs to activate this LogTag carefully, as it prints logs of all failed procedures. This action can cause an overflow of service logs.

   - It prints the following log parameters:
     - Transaction ID
     - Procedure Name
     - Detailed Error: It includes the following:
       - Failure reason or cause
       - Disconnect reason (if applicable)
       - Failure metrics

5. **Call Flow Statistics LogTag**:
   - This process gets tabulated at the end of all procedures.

6. **Core Dump LogTag**:
   - This process initiates the debugging activities for a core dump utility.

7. **Subscriber Monitoring LogTag**:
   - This process initiates the debugging activities for any kind of failures. It also processes the subscriber monitoring based on the subscriber SUPI.

Timesaver For more information, see *Troubleshooting Information* > Logs.

Call Failure Logs

The following are the synopsis of the Call Failure logs to trigger session management policies towards the SMF:

- For call failures and call releases, the disconnect-reason-based statistics are defined and attached.
- Disconnect Reasons—They are classified into three categories as the following:
  - Major
  - Minor
  - Normal
- On call failure or call release scenarios, messages are logged as the following:
  - Major (Catastrophic) Category Disconnect Reasons—The transaction log at Error level is to display the Error category, Disconnect reason, along with Event Trace, and Session keys.
  - Minor (Critical) Category Disconnect Reasons—The transaction log at Warning level to dump the Disconnect reason, along with Event Trace, and Session keys.
  - Normal Category Disconnect Reasons—The transaction log at Debug level to dump the Disconnect reason along with the Event Trace and Session keys.
  - Graceful Core Dump—This transaction log also gets performed with the same LogTag and levels.
- A new LogTag to log the Call Failures also gets defined.
- The Transaction ID gets printed in a Call Failure logs to align with the logs and call flow.
- Based on the disconnect reason statistics, the callfailure tag with the corresponding log level can be enabled to collect further information of the session. The following is a list of examples:
  - Event Trace
  - Session keys
  - Core Dump

Sample Dump for Call Failure Logs

The following is a sample dump for Call Failure logs:

Example: For a procedure lapsed time of over two seconds, a Warning message gets logged with the event trace.

```
2022/08/23 11:10:31.222 [WARN] [smf-service.smf-app.callfailure-debug] 2sec Time Elapsed:
EVENT TRACE SessionKeys[[imsi-123456789012345:5 (pk)]]
```

```
CurIndex:[2], CurProcInst:[0], CreateTimeStamp:[2022-08-23 11:10:31.204 +0530 IST],
BaseTimeStamp:[2022-08-23 11:10:31.204 +0530 IST]
|INDEX|EVENT NAME                          |EVENT TYPE           |PROC NAME
                           |PROC INST |TXN ID    |TIMESTAMP                          |
|1    |N11SmContextCreateReq               |INCOMING_EVENT       |PDU Session Establishment
                           |1         |1         |2022-08-23 11:10:31.215 +0530 IST  |
|2    |N11SmContextCreateReq               |ENDPROC_EVENT        |PDU Session Establishment
                           |1         |1         |2022-08-23 11:10:31.221 +0530 IST  |
```

Configuring the Call Failure Logs

To configure this feature, use the following configuration:

logging name smf-service.smf-app.callfailure-debug level application/transaction <debug level>

NOTES:

- **Warning**—The logging level supported for call failure LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
config
   logging name smf-service.smf-app.callfailure-debug level transaction warn
   logging name smf-service.smf-app.callfailure-debug level application error
exit
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name smf-service.smf-app.callfailure-debug
 logging name smf-service.smf-app.callfailure-debug level application error
 logging name smf-service.smf-app.callfailure-debug level transaction warn
```

Procedure Failure Logs

The following are the synopsis of the Procedure Failure logs to trigger session management policies towards the SMF:

- During the termination scenario, each Procedure Failure log turns on the EndProcedure.
- During the failure scenario, the Error Code returns to the SMF Infra indicating whether the procedure was a success or a failure.
- On failure, a Transaction log (common to all procedures) with level warning is used to dump the session details as the following:
  - Event Trace
  - Procedure Name

  - Error Details—It contains failure reason or cause, disconnect reason (if applicable) and failure metrics.

- The Procedure Failure LogTag is common to all procedures.
- Procedure-specific LogTags will be configured for each procedure and used to enable logging for specific procedure failures.
- The Transaction ID gets printed in a procedure failure logs to align with the logs and call flow.
- The detailed errors note gets printed in the procedure failure logs. It helps in understanding the problem, the actual error cause, disconnect reason (if applicable), and failure metrics.

Logging a Message for Procedure Failure Logs

The following is a sample example of a LogTag for Procedure Failure logs:

```
2023/01/15 11:34:30.002 [WARN] [smf-service.smf-app.pdnsetup-procfailure]
[Txn :7]Procedure=[PDN Connect [LTE]], PduState=[IDLE], Rat-Type=[rat_type_unknown],
FailureReason=[udm_subscribe_notify_failure],
DisconnectReason=[disc_pdnsetup_udm_sub_notify_resp_failed]
2023/01/15 11:34:30.002 [WARN] [smf-service.smf-app.pdnsetup-procfailure] [Txn :7]EVENT
TRACE
CurIndex:[8], CurProcInst:[0], CreateTimeStamp:[2022-09-13 11:34:27.942 +0000 UTC],
BaseTimeStamp:[2022-09-13 11:34:27.942 +0000 UTC]

|INDEX|EVENT NAME |EVENT TYPE |PROC NAME |PROC INST |TXN ID |TIMESTAMP |

|1 |S5S8CreateSessReq |INCOMING_EVENT |PDN Connect [LTE] |1 |7 |2022-09-13 11:34:27.942
+0000 UTC |
|2 |N10RegistrationSuccess |INCOMING_EVENT |PDN Connect [LTE] |1 |7 |2022-09-13 11:34:27.943
 +0000 UTC |
```

Configuring the Procedure Failure Logs

To configure this feature, use the following configuration:

```
logging  name  smf-service.smf-app.pdusetup-procfailure          level
transaction <debug level>
logging  name  smf-service.smf-app.pdnsetup-procfailure   level
transaction <debug level>
logging  name  smf-service.smf-app.pdurelease-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.pdndisconnect-procfailure       level
  transaction <debug level>
logging  name  smf-service.smf-app.5gim-procfailure
 level          transaction <debug level>
logging  name  smf-service.smf-app.xnho-procfailure
 level          transaction <debug level>
logging  name  smf-service.smf-app.n2ho-procfailure
level   transaction <debug level>
logging  name  smf-service.smf-app.nrtowifiho-procfailure
level   transaction <debug level>
```

```
logging  name  smf-service.smf-app.enbtowifiho-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.wifitonrho-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.wifitoenbho-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.4gdedbrr-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.pdnmodmbr-procfailure            level
  transaction <debug level>
logging  name  smf-service.smf-app.5gmodify-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.5g4gho-procfailure
level  transaction <debug level>
logging  name  smf-service.smf-app.n26ho-procfailure
 level          transaction <debug level>
```

NOTES:

- **Warning**—The logging level supported for procedure failure LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
config
  logging name smf-service.smf-app.4gdedbrr-procfailure level transaction warn
  logging name smf-service.smf-app.5g4gho-procfailure level transaction warn
  logging name smf-service.smf-app.5gim-procfailure level transaction warn
  logging name smf-service.smf-app.5gmodify-procfailure level transaction warn
  logging name smf-service.smf-app.enbtowifiho-procfailure level transaction warn
  logging name smf-service.smf-app.n26ho-procfailure level transaction warn
  logging name smf-service.smf-app.n2ho-procfailure level transaction warn
  logging name smf-service.smf-app.nrtowifiho-procfailure level transaction warn
  logging name smf-service.smf-app.pdndisconnect-procfailure level transaction warn
  logging name smf-service.smf-app.pdnmodmbr-procfailure level transaction warn
  logging name smf-service.smf-app.pdnsetup-procfailure level transaction warn
  logging name smf-service.smf-app.pdurelease-procfailure level transaction warn
  logging name smf-service.smf-app.pdusetup-procfailure level transaction warn
  logging name smf-service.smf-app.wifitoenbho-procfailure level transaction warn
  logging name smf-service.smf-app.wifitonrho-procfailure level transaction warn
  logging name smf-service.smf-app.xnho-procfailure level transaction warn
exit
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name
 logging name smf-service.smf-app.4gdedbrr-procfailure level transaction warn
 logging name smf-service.smf-app.5g4gho-procfailure level transaction warn
 logging name smf-service.smf-app.5gim-procfailure level transaction warn
 logging name smf-service.smf-app.5gmodify-procfailure level transaction warn
 logging name smf-service.smf-app.enbtowifiho-procfailure level transaction warn
 logging name smf-service.smf-app.n26ho-procfailure level transaction warn
 logging name smf-service.smf-app.n2ho-procfailure level transaction warn
 logging name smf-service.smf-app.nrtowifiho-procfailure level transaction warn
```

```
logging name smf-service.smf-app.pdndisconnect-procfailure level transaction warn
logging name smf-service.smf-app.pdnmodmbr-procfailure level transaction warn
logging name smf-service.smf-app.pdnsetup-procfailure level transaction warn
logging name smf-service.smf-app.pdurelease-procfailure level transaction warn
logging name smf-service.smf-app.pdusetup-procfailure level transaction warn
logging name smf-service.smf-app.wifitoenbho-procfailure level transaction warn
logging name smf-service.smf-app.wifitonrho-procfailure level transaction warn
logging name smf-service.smf-app.xnho-procfailure level transaction warn
```

Generic Procedure Failure Logs

The following are the synopsis of the Generic Procedure Failure logs to trigger session management policies towards the SMF:

- During the termination scenario, each Procedure Failure log turns on the EndProcedure.
- During the failure scenario, the Error Code returns to the SMF Infra indicating whether the procedure was a success or a failure.
- On failure, a Generic Transaction log (common to all procedures) with level warning is used to dump the session details as the following:
  - Transaction ID
  - Procedure Name
  - Detailed Error—It contains failure reason or cause, disconnect reason (if applicable) and failure metrics.
- The Generic Procedure Failure LogTag is common to all procedures.
- When a specific procedure failure LogTag is disabled and a Generic Procedure Failure LogTag is enabled, then the SMF doesn't print the Event Trace.
- The Event Trace is printed, only when a specific procedure failure LogTag gets enabled.
- The Transaction ID gets printed in a procedure failure logs to align with the logs and call flow.
- The Detailed errors note gets printed in the generic procedure failure logs. It helps in understanding the problem, the actual error cause, disconnect reason (if applicable), and failure metrics.

Logging a Message for Generic Procedure Failure Logs

The following is a sample example of a LogTag for Generic Procedure Failure logs:

```
2023/01/15 10:58:01.171 [WARN] [smf-service.smf-app.procfailure] [Txn :1]Procedure=[PDN
Connect [LTE]],
PduState=[IDLE], Rat-Type=[rat_type_unknown], FailureReason=[udm_subscribe_notify_failure],

DisconnectReason=[disc_pdnsetup_udm_sub_notify_resp_failed]
```

Configuring the Generic Procedure Failure Logs

To configure this feature, use the following configuration:

logging name smf-service.smf-app.procfailure level transaction <debug level>

NOTES:

- **Warning**—The logging level supported for generic procedure failure LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
config
 logging name smf-service.smf-app.procfailure level transaction warn
exit
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name smf-service.smf-app.procfailure
logging name smf-service.smf-app.procfailure level transaction warn
```

Additional Call Flow Failure Logs

The following are the synopsis of the Additional Call Flow Failure logs to trigger session management policies towards the SMF:

- Extra LogTags are added for specific procedures.
- These LogTags can be enabled for specific conditions between the procedure flow.
- The Additional Call Flow Failure logs help in identifying a procedure lapsed event and reciprocate with a logged warning message in the event trace.

Sample Dump for Additional Call Flow Failure Logs

The following is a sample dump for Additional Call Flow Failure logs:

Example: For a procedure lapsed time of over two seconds, a warning message gets logged with the event trace.

```
2022/08/23 11:10:31.222 [WARN] [smf-service.smf-app.5gmodify-failure] 2sec Time Elapsed:
EVENT TRACE SessionKeys[[imsi-123456789012345:5 (pk)]]
CurIndex:[2], CurProcInst:[0], CreateTimeStamp:[2022-08-23 11:10:31.204 +0530 IST],
BaseTimeStamp:[2022-08-23 11:10:31.204 +0530 IST]
|INDEX|EVENT NAME                          |EVENT TYPE           |PROC NAME
                           |PROC INST |TXN ID    |TIMESTAMP                          |
```

Logging a Message for Additional Call Flow Failure Logs

The following is a sample example of a LogTag for Additional Call Flow Failure logs:

```
2022/08/23 11:10:31.222 [WARN] [smf-service.smf-app.5gmodify-failure] 2sec Time Elapsed:
EVENT TRACE SessionKeys[[imsi-123456789012345:5 (pk)]]
CurIndex:[2], CurProcInst:[0], CreateTimeStamp:[2022-08-23 11:10:31.204 +0530 IST],
BaseTimeStamp:[2022-08-23 11:10:31.204 +0530 IST]
|INDEX|EVENT NAME                          |EVENT TYPE           |PROC NAME
                          |PROC INST |TXN ID    |TIMESTAMP                          |
```

Configuring the Additional Call Flow Failure Logs

To configure this feature, use the following configuration:

```
Logging name smf-service.smf-app.4gdedbrr-failure level transaction <debug
 level>
logging      name     smf-service.smf-app.pdnmodmbr-failure     level
  transaction     <debug level>
logging      name     smf-service.smf-app.5gmodify-failure          level
     transaction     <debug level>
logging      name     smf-service.smf-app.5g4gho-failure            level
     transaction     <debug level>
logging      name     smf-service.smf-app.n26ho-failure             level
     transaction     <debug level>
logging      name     smf-service.smf-app.wifi-nr-ho
level     transaction     <debug level>
logging      name     smf-service.smf-app.datacheck       level
transaction     <debug level>
logging      name     smf-service.smf-app.N16             level
transaction     <debug level>
logging      name     smf-service.smf-app.erir             level
transaction     <debug level>
logging      name     smf-service.smf-app.flagdb            level
transaction     <debug level>
logging     name     smf-service.smf-app.dcr        level     transaction
     <debug level>
logging      name     smf-service.smf-app.dcr-ue        level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-dup       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-mme       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-pcf       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-ubr       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-absent      level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-amf       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-chf      level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-gnb       level
transaction     <debug level>
```

```
logging      name     smf-service.smf-app.dcr-brr-smf       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-udm       level
transaction     <debug level>
logging      name     smf-service.smf-app.dcr-brr-ue       level
transaction     <debug level>
logging      name     smf-service.smf-app.epsfb        level
transaction     <debug level>
```

NOTES:

- **Warning**—The logging level supported for additional call flow failure LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
config
  logging name smf-service.smf-app.4gdedbrr-failure level transaction warn
  logging name smf-service.smf-app.5g4gho-failure level transaction warn
  logging name smf-service.smf-app.5gmodify-failure level transaction warn
  logging name smf-service.smf-app.N16 level transaction warn
  logging name smf-service.smf-app.datacheck level transaction warn
  logging name smf-service.smf-app.dcr level transaction warn
  logging name smf-service.smf-app.dcr-brr-absent level transaction warn
  logging name smf-service.smf-app.dcr-brr-amf level transaction warn
  logging name smf-service.smf-app.dcr-brr-chf level transaction warn
  logging name smf-service.smf-app.dcr-brr-dup level transaction warn
  logging name smf-service.smf-app.dcr-brr-gnb level transaction warn
  logging name smf-service.smf-app.dcr-brr-mme level transaction warn
  logging name smf-service.smf-app.dcr-brr-pcf level transaction warn
  logging name smf-service.smf-app.dcr-brr-smf level transaction warn
  logging name smf-service.smf-app.dcr-brr-ubr level transaction warn
  logging name smf-service.smf-app.dcr-brr-udm level transaction warn
  logging name smf-service.smf-app.dcr-brr-ue level transaction warn
  logging name smf-service.smf-app.dcr-ue level transaction warn
  logging name smf-service.smf-app.epsfb level transaction warn
  logging name smf-service.smf-app.erir level transaction warn
  logging name smf-service.smf-app.flagdb level transaction warn
  logging name smf-service.smf-app.n26ho-failure level transaction warn
  logging name smf-service.smf-app.pdnmodmbr-failure level transaction warn
  logging name smf-service.smf-app.wifi-nr-ho level transaction warn
end
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name
 logging name smf-service.smf-app.4gdedbrr-failure level transaction warn
 logging name smf-service.smf-app.5g4gho-failure level transaction warn
 logging name smf-service.smf-app.5gmodify-failure level transaction warn
 logging name smf-service.smf-app.N16 level transaction warn
 logging name smf-service.smf-app.datacheck level transaction warn
 logging name smf-service.smf-app.dcr level transaction warn
 logging name smf-service.smf-app.dcr-brr-absent level transaction warn
 logging name smf-service.smf-app.dcr-brr-amf level transaction warn
 logging name smf-service.smf-app.dcr-brr-chf level transaction warn
 logging name smf-service.smf-app.dcr-brr-dup level transaction warn
```

```
logging name smf-service.smf-app.dcr-brr-gnb level transaction warn
logging name smf-service.smf-app.dcr-brr-mme level transaction warn
logging name smf-service.smf-app.dcr-brr-pcf level transaction warn
logging name smf-service.smf-app.dcr-brr-smf level transaction warn
logging name smf-service.smf-app.dcr-brr-ubr level transaction warn
logging name smf-service.smf-app.dcr-brr-udm level transaction warn
logging name smf-service.smf-app.dcr-brr-ue level transaction warn
logging name smf-service.smf-app.dcr-ue level transaction warn
logging name smf-service.smf-app.epsfb level transaction warn
logging name smf-service.smf-app.erir level transaction warn
logging name smf-service.smf-app.flagdb level transaction warn
logging name smf-service.smf-app.n26ho-failure level transaction warn
logging name smf-service.smf-app.pdnmodmbr-failure level transaction warn
logging name smf-service.smf-app.wifi-nr-ho level transaction warn
```

Event Trace Logs

The following are the synopsis of the Event Trace logs to trigger session management policies towards the SMF:

- Event Trace logs provide an execution sequence for a session. It represents the following:
  - Message Type
  - Event Type—Incoming or Outgoing or Internal submitted event type.
  - Procedure Type and Procedure Instance—Where the message is associated with.
  - Txn ID—Where the message is associated with.
  - Timestamp—For the same message associated with
- Event Trace logs also describe other events as the following:
  - Local2DB—When the session details are derived into a primitive structure.
  - DB2Local—When the session details aren't derived from DB to Session DS.
  - Procedure Events—When it conveys the END or SUSPEND or ABORT or CLEANUP events also.

Sample Dump for Event Trace Logs

The following is a sample dump for Event Trace logs:

```
2020/09/10 13:16:16.177 smf-service [DEBUG] [Genericutil.go:5739]
[smf-service0.smf-app.event-trace] EVENT TRACE SessionKeys[[imsi-123456789012345:5 (pk)]]
CurIndex:[31], CurProcInst:[1], CreateTimeStamp:[2020-09-10 13:16:15.503 +0000 UTC],
BaseTimeStamp:[2020-09-10 13:16:15.503 +0000 UTC]
|INDEX|EVENT NAME                          |EVENT TYPE           |PROC NAME
                          |PROC INST |TXN ID    |TIMESTAMP                          |

|1    |N11SmContextCreateReq               |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.503 +0000 UTC  |
|2    |N10RegistrationRequest              |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.525 +0000 UTC  |
|3    |N10RegistrationSuccess              |INCOMING_EVENT       |PDU Session Establishment
```

```
                            |1          |1776      |2020-09-10 13:16:15.605 +0000 UTC  |
|4    |N10SubscriptionFetchReq            |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1776      |2020-09-10 13:16:15.606 +0000 UTC  |
|5    |N10SubscriptionFetchSuccess        |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1776      |2020-09-10 13:16:15.644 +0000 UTC  |
|6    |N10SubscribeForNotificationReq     |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1776      |2020-09-10 13:16:15.65 +0000 UTC   |
|7    |N10SubscribeForNotificationSuccess |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1776      |2020-09-10 13:16:15.67 +0000 UTC   |
|8    |NIntSelfTxnPduSetup                |INTERNAL_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.67 +0000 UTC   |
|9    |Message Type None                  |LOCAL_2_DB           |Unknown
                            |0          |0         |2020-09-10 13:16:15.671 +0000 UTC  |
|10   |N11SmContextCreateSuccess          |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1776      |2020-09-10 13:16:15.711 +0000 UTC  |
|11   |NIntSelfTxnPduSetup                |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.712 +0000 UTC  |
|12   |N7SmPolicyCreateReq                |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.712 +0000 UTC  |
|13   |N7SmPolicyCreateSuccess            |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.797 +0000 UTC  |
|14   |NmgrRersourceMgmtRequest           |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.805 +0000 UTC  |
|15   |NmgrRersourceMgmtResponse          |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.814 +0000 UTC  |
|16   |N7SmPolicyUpdateReq                |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.823 +0000 UTC  |
|17   |N7SmPolicyUpdateSuccess            |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.853 +0000 UTC  |
|18   |N40ChargingDataReq                 |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.854 +0000 UTC  |
|19   |N40ChargingDataSuccess             |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.912 +0000 UTC  |
|20   |N4SessionEstablishmentReq          |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.921 +0000 UTC  |
|21   |N4SessionEstablishmentSuccess      |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.986 +0000 UTC  |
|22   |N11EbiAssignmentReq                |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.99 +0000 UTC   |
|23   |N11EbiAssignmentRsp                |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:16.017 +0000 UTC  |
|24   |N11N1N2MessageTransferReq          |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:16.021 +0000 UTC  |
|25   |N11N1N2MessageTransferSuccess      |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:16.096 +0000 UTC  |
|26   |Message Type None                  |LOCAL_2_DB           |Unknown
                            |0          |0         |2020-09-10 13:16:16.096 +0000 UTC  |
|27   |N11SmContextUpdateReq              |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.111 +0000 UTC  |
|28   |N4SessionModificationReq           |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.114 +0000 UTC  |
|29   |N4SessionModificationSuccess       |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.155 +0000 UTC  |
|30   |N4GtpuRouterAdvertisementReq       |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.156 +0000 UTC  |
```

```
|31   |N4SessionModificationSuccess       |ENDPROC_EVENT        |PDU Session Establishment
                          |1         |1778      |2020-09-10 13:16:16.175 +0000 UTC  |
```

Logging a Message for Event Trace Logs

The following is a sample example of a LogTag for Event Trace logs:

```
2020/09/10 13:16:16.177 smf-service [DEBUG] [Genericutil.go:5739]
[smf-service0.smf-app.event-trace] EVENT TRACE SessionKeys[[imsi-123456789012345:5 (pk)]]
CurIndex:[31], CurProcInst:[1], CreateTimeStamp:[2020-09-10 13:16:15.503 +0000 UTC],
BaseTimeStamp:[2020-09-10 13:16:15.503 +0000 UTC]
|INDEX|EVENT NAME                         |EVENT TYPE           |PROC NAME
                          |PROC INST |TXN ID    |TIMESTAMP                          |

|1    |N11SmContextCreateReq              |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.503 +0000 UTC  |
|2    |N10RegistrationRequest             |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.525 +0000 UTC  |
|3    |N10RegistrationSuccess             |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.605 +0000 UTC  |
|4    |N10SubscriptionFetchReq            |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.606 +0000 UTC  |
|5    |N10SubscriptionFetchSuccess        |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.644 +0000 UTC  |
|6    |N10SubscribeForNotificationReq     |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.65 +0000 UTC   |
|7    |N10SubscribeForNotificationSuccess |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.67 +0000 UTC   |
|8    |NIntSelfTxnPduSetup                |INTERNAL_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.67 +0000 UTC   |
|9    |Message Type None                  |LOCAL_2_DB           |Unknown
                          |0         |0         |2020-09-10 13:16:15.671 +0000 UTC  |
|10   |N11SmContextCreateSuccess          |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1776      |2020-09-10 13:16:15.711 +0000 UTC  |
|11   |NIntSelfTxnPduSetup                |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.712 +0000 UTC  |
|12   |N7SmPolicyCreateReq                |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.712 +0000 UTC  |
|13   |N7SmPolicyCreateSuccess            |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.797 +0000 UTC  |
|14   |NmgrRersourceMgmtRequest           |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.805 +0000 UTC  |
|15   |NmgrRersourceMgmtResponse          |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.814 +0000 UTC  |
|16   |N7SmPolicyUpdateReq                |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.823 +0000 UTC  |
|17   |N7SmPolicyUpdateSuccess            |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.853 +0000 UTC  |
|18   |N40ChargingDataReq                 |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.854 +0000 UTC  |
|19   |N40ChargingDataSuccess             |INCOMING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.912 +0000 UTC  |
|20   |N4SessionEstablishmentReq          |OUTGOING_EVENT       |PDU Session Establishment
                          |1         |1777      |2020-09-10 13:16:15.921 +0000 UTC  |
|21   |N4SessionEstablishmentSuccess      |INCOMING_EVENT       |PDU Session Establishment
```

```
                            |1          |1777      |2020-09-10 13:16:15.986 +0000 UTC  |
|22   |N11EbiAssignmentReq                |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:15.99 +0000 UTC   |
|23   |N11EbiAssignmentRsp                |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:16.017 +0000 UTC  |
|24   |N11N1N2MessageTransferReq          |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:16.021 +0000 UTC  |
|25   |N11N1N2MessageTransferSuccess      |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1777      |2020-09-10 13:16:16.096 +0000 UTC  |
|26   |Message Type None                  |LOCAL_2_DB           |Unknown
                            |0          |0         |2020-09-10 13:16:16.096 +0000 UTC  |
|27   |N11SmContextUpdateReq              |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.111 +0000 UTC  |
|28   |N4SessionModificationReq           |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.114 +0000 UTC  |
|29   |N4SessionModificationSuccess       |INCOMING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.155 +0000 UTC  |
|30   |N4GtpuRouterAdvertisementReq       |OUTGOING_EVENT       |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.156 +0000 UTC  |
|31   |N4SessionModificationSuccess       |ENDPROC_EVENT        |PDU Session Establishment
                            |1          |1778      |2020-09-10 13:16:16.175 +0000 UTC  |
```

Configuring the Event Trace Logs

To configure this feature, use the following configuration:

logging name smf-service.smf-app.event-trace level transaction debug-level

NOTES:

- **Warning | Error**—The logging level supported for Event Trace LogTags.
- **Session Keys | Procedure Name | Event Trace**—The logging level parameters supported for Event Trace LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
logging name smf-service.smf-app.event-trace level transaction warn
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name smf-service.smf-app.event-trace
logging name smf-service.smf-app.event-trace level transaction warn
```

Call Flow Statistics Logs

The following are the synopsis of the Call Flow Statistics Logs to trigger session management policies towards the SMF:

- Currently, each call flow has the following general labels and other specific labels:
  - Attempted
  - Success
  - Failure + Reason
- All the call flow statistics gets integrated with EndProcedure.
- Each DispositionEnd used to end the call flow in the procedure, concludes either as a success or a failure scenario. Each failure scenario has a reason for integration as well.
- Failure causes can be classified into different levels. They are further mapped and classified as the following:
  - Major or Error
    - Internal Error—All internal errors due to which the call flow is failing, are categorized as error.
    - Transaction Timeout Error—On interfaces, outbound response timeouts (transaction timeouts) are categorized as errors.
  - Minor or Warning
    - Server Error—On interfaces, outbound responses with the http status code 5xx (server errors) are categorized as a warning.
    - Client Error—On interfaces, inbound responses with the http status code 4xx (client errors) or timeouts are categorized as a warning.
  - Normal
    - Server Error—On interfaces, inbound responses with the http status code 5xx (server errors) are categorized as normal.
    - Client Error—On interfaces, outbound responses with http status code 4xx (client errors) are categorized as normal.
- Unknown—Those scenarios, where the call flow gets failed, but the reason doesn't get populated, fall under the unknown cause.
- Further, these scenarios can be enhanced to cases, other statistics labels of statistics, which get integrated to empty statuses.

  Example: RAT type integrated as empty, DNN type not integrated, and so on.

Configuring the Call Flow Statistics Logs

To configure this feature, use the following configuration:

logging name smf-service.smf-app.stats-debug level transaction debug-level

NOTES:

- **Warning | Error**—The logging level supported for call flow statistics LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
logging name smf-service.smf-app.stats-debug level transaction warn
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name smf-service.smf-app.stats-debug
logging name smf-service.smf-app.stats-debug level transaction warn
```

Core Dump Utility Logs

The following are the synopsis of the Core Dump Utility Logs to trigger session management policies towards the SMF:

- The Core Dump Utility logs provide functionality to conditionally generate a core dump for debugging purposes.
- The core file generation doesn't kill the process and it continues to execute as usual.
- The core dump functionality is disabled by default and it needs to be enabled by specifying the appropriate configurable parameters.

Important The core dump generation is a performance impacting activity. Hence, it's recommended to restrict the core dump to a minimum number, such as one or two core dumps for every 15 minutes. As a result, you need to fine-tune the core dump, if you see any performance impact.

Logging a Message for Core Dump Utility Logs

The following is a sample example of a LogTag for Core Dump Utility Logs:

```
2022/10/14 09:14:38.809 [WARN] [smf-service.smf-app.pdusetup-procfailure] [Txn Id: 3] {PDU
 Session Establishment} -> COREDUMP ->
Generating Core: current core count 1,
File:/opt/workspace/smf-service/src/smf-service/procedures/pdusetup/procedure.go:612
2022/10/14 09:15:38.817 [WARN] [smf-service.smf-app.pdusetup-procfailure] {PDU Session
```

```
Establishment} -> COREDUMP ->
Generating Core: current core count 2,
File:/opt/workspace/smf-service/src/smf-service/procedures/generic/CallFailureDebug.go:70
```

Configuring the Core Dump Utility Logs

To configure this feature, use the following configuration:

```
config
   dump core [ count count_number | interval interval_details | expires
expires_details | pod-name pod_name | file-detail file_detail ]
   end
```

NOTES:

- **count** *count_number*—Specify the maximum number of times the core dump can be taken. Example: Two core dumps can be taken in a span of 15 minutes, where two is the count. Range: 0-50
- **interval** *interval_details*—Specify the total duration of the interval (in minutes) to take the core dumps. Example: Two cores can be taken in a span of 15 minutes, where 15 minutes is the total interval time to take two core dumps. Range: 1-3600
- **expires** *expires_details*—Specify the time after which a core agent stops generating a core dump. Format: CCYY-MM-DDTHH:MM:SS, with an example: 2020-03-24T23:15:00+05:30 or 2022-10-17T19:00:00+00:00
- **pod-name** *pod_name*—List the name of the pod to enable core dump activities.
- **file-detail** *file_detail*—List the file name, line number to a specific core dump. Example: Procedures or PDUIM or procedure.go:1902, the maximum size is 10.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
dump core count 2 interval 15 expires pod-name [ smf-service-n0-0 smf-service-n0-1 ]
file-detail [ procedures/generic/CallFailureDebug.go:70 procedures/pdusetup/procedure.go:612
 ]
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config dump core
dump core
count 2 interval 15 expires 2023-01-18T18:00:00-00:00 pod-name [ smf-service-n0-0
smf-service-n0-1 ] file-detail [ procedures/generic/CallFailureDebug.go:70
procedures/pdusetup/procedure.go:612 ]
exit
```

DNN Profile Optimization

Configuring common attributes multiple times in a different DNN profile is a repetitive activity for the Operators. It becomes time consuming activity as the number of DNN profiles gets increased.

SMF provides a way to configure common attributes in a parent DNN profile template and reuse it in other DNN profiles, for example child DNN profiles as and when required.

Note The DNN Profile Optimization feature is currently qualified only for the SMF and not for the cnSGW.

How it Works

This section provides details about how the network operators can perform the DNN profile configuration to inherit or reuse a parent DNN profile:

Table 5: Preparation to Configure the Parent DNN Profile

Step	Description
1	Configure parent DNN profile having common attributes.
2	Inherit the configured parent DNN profile in a child DNN profile. The following conditions apply: • If a child profile is trying to inherit a DNN Profile that doesn't exists, then the SMF CLI displays an error while committing the changes. • If the parent DNN profile is deleted but found in a child profile, then SMF CLI displays an error while committing the changes. **Note** Delink the Parent profile from the Child DNN profile to delete the Parent DNN profile. • If the attributes configured in a child DNN profile do not belong to any parent DNN profile, then the operator can modify those attributes in an individual child DNN profile.

Step	Description
3	

Step	Description

	SMF gets request for the child DNN profile. If the child DNN profile has an *inherit* attribute, then SMF resolves the inherited/parent profile on the following conditions:
	• If the parent profile is found and the child profile doesn't override parent attributes, then SMF updates the child profile with inherited attributes.
	• If a Parent profile is found and a Child profile overrides some or all the parent attributes, then SMF prioritizes overridden attributes over a Parent profile and updates the child profile by taking overridden attributes from a Child profile and the rest of the attributes from a Parent profile.
	• If a Parent profile has attributes in the form of arrays or slices, and the Child profile also has the same attributes configured, then the resultant DNN profile overrides that attribute completely from the Child profile, which means the combined profile contains the value of the Child attribute.
	Example for DNN Profiles:
	`profile dnn intershat_Parent // parent dnn profile`
	`ims mark qci [ 83 128 ]`
	`network-element-profiles chf chf1`
	`network-element-profiles amf amf1`
	`network-element-profiles pcf pcf1`
	`network-element-profiles udm udm1`
	`charging-profile chgprfFBC`
	`override [ charging-profile charging-characteristics-id ]`
	`virtual-mac b6:6d:47:47:47:47`
	`session type IPV4 allowed [ IPV6 IPV4V6 ]`
	`dcnr true`
	`exit`
	`profile dnn intershatRoamer // child dnn profile`
	`inherit intershat_Parent`
	`override [ charging-characteristics-id charging-qbc-profile ]`
	`exit`
	Example for Resultant Profiles:
	`profile dnn intershatRoamer // Resultant dnn profile`
	`ims mark qci [ 83 128 ]`
	`network-element-profiles chf chf1`
	`network-element-profiles amf amf1`
	`network-element-profiles pcf pcf1`
	`network-element-profiles udm udm1`
	`charging-profile chgprfFBC`
	`override [charging-characteristics-id charging-qbc-profile ]`
	`virtual-mac b6:6d:47:47:47:47`
	`session type IPV4 allowed [ IPV6 IPV4V6 ]`
	`dcnr true`
	`exit`
	• If a Parent profile has attributes in form of structures and the Child profile also has the same attributes configured, then the resultant DNN profile overrides that attribute completely from child, which means the combined profile contains the value of a Child attribute.
	Example for DNN Profiles:
	`profile dnn intershat_Parent // parent dnn profile`
	`charging-profile chgprfFBC`
	`virtual-mac b6:6d:47:47:47:47`

Step	Description
	dcnr true network-element-profiles chf chf1 network-element-profiles amf amf1 network-element-profiles udm udm1 exit profile dnn intershatRoamer // child dnn profile inherit intershat_Parent network-element-profiles pcf pcf1 exit **Example for Resultant profiles**: profile dnn intershatRoamer // Resultant dnn profile charging-profile chgprfFBC virtual-mac b6:6d:47:47:47:47 dcnr true network-element-profiles pcf pcf1 exit **Note** When the SMF gets a request for the Child DNN profile and the child DNN profile doesn't have an *inherit* attribute, then the SMF returns all attributes from the child DNN profile.
4	Update the parent and child DNN profile at runtime. After changes are committed, SMF picks the latest attributes from the Child and Parent profiles.

Configuring Parent DNN Profiles

Use the following CLI command to configure parent DNN profiles having common attributes. you can define all the common attributes in a separate DNN profile and then use this parent profile in other DNN profiles by using the new CLI option.

SMF fetches the inherited DNN profile, if found then updates the child DNN profile(s) with the attributes of inherited DNN profile.

```
config
  profile dnn dnn_profile_name  inherit dnn_template_name
     mode dnn_mode
       dns primary ipv4 addressipv4_address
       dns primary ipv6 addressipv6_address
       dns secondary ipv4 address ipv4_address
       dns secondary ipv6 addressipv6_address
       network-element-profiles {  amf | chf | pcf | udm  } nf_profile_name
 exit
profile dnn dnn_profile_name
   inherit dnn_template_name
   dnn rmgr  ims_pool_ipv6
   exit
```

NOTES:

- **profile dnn** *dnn_profile_name* **inherit** *dnn_template_name*: Specify the DNN profile and specify the DNN template name to inherit to a child profile.

- **mode** *dnn_mode*: Specify the DNN mode of operation.
- **network-element-profiles { amf | chf | ocs | ofcs | pcf | pcrf | sepp | smf | udm }** *nf_profile_name*: Specify one or more NF types, such as AMF, CHF, OCS, OFCS, PCF, PCRF, SEPP, SMF and UDM as the network element profile. nf_profile_name must be an alphanumeric string representing the corresponding network element profile name.
  - This is an optional configuration. By default, this CLI command is disabled.
  - You can configure multiple profiles within a given service.
  - To disable the configuration, use the no network-element-profiles { amf | chf | ocs | ofcs | pcf | pcrf | sepp | smf | udm } nf_profile_name command.

Configuration Example

The following is an example configuration.

```
profile dnn dnnprof-ims.epdg.prod
  dns primary ipv4 10.177.0.34
  dns primary ipv6 fd00:976a::9
  dns secondary ipv4 10.177.0.210
  dns secondary ipv6 fd00:976a::10
  network-element-profiles chf nfprf-chf1
  network-element-profiles amf nfprf-amf1
  network-element-profiles pcf nfprf-pcf1
  network-element-profiles udm nfprf-udm1
  dnn ims.epdg.prod network-function-list [ chf pcf udm upf ]
  timeout up-idle 3600 cp-idle 7320
  charging-profile           chgprof-1
  pcscf-profile              pcscf1
  ppd-profile                ppd-prof1
  ssc-mode 1 allowed [ 2 ]
  session type IPV6
  session skip-ind false
  upf apn ims.epdg.prod
  qos-profile                5qi-to-dscp-mapping-table-IMS
  always-on                  true
  dcnr                       true
  userplane-inactivity-timer 3600
  only-nr-capable-ue         false
exit

// Child DNN profile
profile dnn dnnprof-ims.prod
  inherit dnnprof-ims.epdg.prod
  dnn rmgr ims-pool-ipv6
exit
```

Monitor Subscriber (MonSub) Logs

The following are the synopsis of the Monitor Subscriber (MonSub) Logs to trigger session management policies towards the SMF:

- The logs help to control the logging level of transaction logs, when the monitor subscriber CLI gets enabled.
- The Monitor Subscriber CLI captures the transaction logs for a given SUPI or IMSI.

- The Monitor Subscriber Logger uses a specific logging level, used for the subscriber, for which the monitor subscriber CLI gets triggered.
- The SMF must support per-subscriber monitoring activities. If required, it must be used to monitor a particular subscriber based on the appropriate configurable parameters, such as SUPI or IMSI.
- As a default value, the monitor subscriber functionality is disabled. It gets enabled manually by specifying the appropriate configurable parameters.

Logging a Message for MonSub Logs

The following is a sample example of a LogTag for MonSub Logs:

```
smf# monitor subscriber imsi 123456789012345 capture-duration 3600 transaction-logs yes
internal-messages yes
Fri Oct 14 09:05:53.902 UTC+00:00
supi: imsi-123456789012345
captureDuration: 3600
enableInternalMsg: true
enableTxnLog: true
namespace(deprecated. Use nf-service instead.): none
nf-service: none
gr-instance: 0
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 318 100 116 100 202 12888 22444 --:--:-- --:--:-- --:--:-- 35333
Command: --header Content-type:application/json --request POST --data

{"commandname":"mon_sub",
"parameters":{"supi":"imsi-123456789012345",
"duration":3600,"enableTxnLog":true,"enableInternalMsg":true,
"action":"start","namespace":"none","nf-service":"none",
"grInstance":0}} http://oam-pod:8879/commands
Result start mon_sub,
fileName
->logs/monsublogs/none.imsi-123456789012345_WithTxnLogs_TS_2022-10-14T09:05:53.926268924.txt
Starting to tail the monsub messages from
file:
logs/monsublogs/none.imsi-123456789012345_WithTxnLogs_TS_2022-10-14T09:05:53.926268924.txt
Defaulted container "oam-pod" out of: oam-pod, apache
Transaction Log received from Instance: SMF.nodemgr.DC.SMF.1
************* TRANSACTION: 00166 *************


TRANSACTION SUCCESS:
Start Time : 2022/10/14 09:07:24.449
GR Instance ID : 1
Txn Type : GtpcAssocEstReq(2114)
Priority : 1
Session Namespace : none(0)
CDL Slice Name : 1
LOG MESSAGES:
2022/10/14 09:07:24.449 [TRACE] [infra.message_log.core] >>>>>>>>
IPC message
Name: GtpcAssocEstReq
```

```
MessageType: GtpcAssocEstReq
Key:
--body--


{"IPv4address":167844075,"restart_counter":65535,
"nodemgr_instance_id":1,"timestamp":1665738444,
"gtpcPathStatus":2,"gtpcInterfaceType":4,"ddnInfo":{},
"gr_instance_id":1,"isNodeStarted":true}
2022/10/14 09:07:24.449 [DEBUG] [nodemgr1.app.Int] GetSessionNamespace for txn id: 166,
Type: 2114
2022/10/14 09:07:24.449 [DEBUG] [nodemgr.gtpmgr.gtp] Received New GTP Peer info
[&GtpcAssocEstReq{IPv4Address:167844075,IPv6Address:[],SupportedFeatures:0,RestartCounter:65535,
NodemgrInstanceId:1,Timestamp:1665738444,OverloadControl:nil,GtpcPathStatus:GTPC_PATH_UP,
GtpcInterfaceType:4,DdnInfo:&DdnInfo{DelayValid:false,ThrottleValRcvd:false,ThrottleActive:false,
DelayValue:0,ThrottleDelayValue:0,ThrottleDelayUnit:0,ThrottleFactor:0,},GrInstanceId:1,
IsNodeStarted:true,SelfRCVal:0,PeerType:0,DeletedAt:0,}]
2022/10/14 09:07:24.449 [INFO] [nodemgr.gtpmgr.gtp] GTPC Path Mgmt Disabled for interface
[4]
2022/10/14 09:07:24.449 [INFO] [nodemgr.gtpmgr.gtp] Rcvd timestamp 1665738444, stored
timestamp 1665738177
peer 10.1.24.235
2022/10/14 09:07:24.449 [DEBUG] [nodemgr.gtpmgr.gtp] Assoc req for already present peer
[10.1.24.235]
2022/10/14 09:07:24.449 [DEBUG] [nodemgr.gtpmgr.gtp] stop timer for existing gtp-peer
[10.1.24.235]
2022/10/14 09:07:24.451 [TRACE] [infra.message_log.core] <<<<<<<<


*******************************************
Transaction Log received from Instance: SMF.nodemgr.DC.SMF.1
************* TRANSACTION: 00167 *************
TRANSACTION SUCCESS:
Start Time : 2022/10/14 09:07:34.061
GR Instance ID : 1
Txn Type : GtpcAssocEstReq(2114)
Priority : 1
Session Namespace : none(0)
CDL Slice Name : 1
LOG MESSAGES:
2022/10/14 09:07:34.062 [TRACE] [infra.message_log.core] >>>>>>>>
IPC message
Name: GtpcAssocEstReq
MessageType: GtpcAssocEstReq
Key:
--body--

{"IPv4address":167844075,"restart_counter":100,"timestamp":1665738454,"gtpcInterfaceType":4,
"ddnInfo":{},"isNodeStarted":true}
2022/10/14 09:07:34.062 [DEBUG] [nodemgr1.app.Int] GetSessionNamespace for txn id: 167,
Type: 2114
2022/10/14 09:07:34.062 [DEBUG] [nodemgr.gtpmgr.gtp] Received New GTP Peer info
[&GtpcAssocEstReq{IPv4Address:167844075,IPv6Address:[],SupportedFeatures:0,RestartCounter:100,
NodemgrInstanceId:0,Timestamp:1665738454,OverloadControl:nil,GtpcPathStatus:GTP_PATH_INVALID,
GtpcInterfaceType:4,DdnInfo:&DdnInfo{DelayValid:false,ThrottleValRcvd:false,ThrottleActive:false,
DelayValue:0,ThrottleDelayValue:0,ThrottleDelayUnit:0,ThrottleFactor:0,},GrInstanceId:0,
IsNodeStarted:true,SelfRCVal:0,PeerType:0,DeletedAt:0,}]
```

```
2022/10/14 09:07:34.062 [INFO] [nodemgr.gtpmgr.gtp] GTPC Path Mgmt Disabled for interface
[4]
2022/10/14 09:07:34.062 [INFO] [nodemgr.gtpmgr.gtp] Rcvd timestamp 1665738454,
stored timestamp 1665738444 peer 10.1.24.235
2022/10/14 09:07:34.062 [DEBUG] [nodemgr.gtpmgr.gtp] Assoc req for already present peer
[10.1.24.235]
2022/10/14 09:07:34.062 [DEBUG] [nodemgr.gtpmgr.gtp] stop timer for existing gtp-peer
[10.1.24.235]
2022/10/14 09:07:34.065 [TRACE] [infra.message_log.core] <<<<<<<<

*******************************************
```

Configuring the Monitor Subscriber Logs

To configure this feature, use the following configuration:

```
config
   monitor subscriber [ supi supi_id | imsi imsi_value | imei imei_id |
capture-duration capture_duration | dump dump_name | gr-instance gr_instance_id |
 internal-messages internal_messages | list list_details | namespace namespace_details
 | nf-service nf_service_details | transaction-logs transaction_logs ]
   end
```

NOTES:

- **supi** *supi_id*—Specify the subscriber identifier. For example, imsi-123456789, imsi-123*
- **imsi** *imsi_value*—Specify the subscriber IMSI. For example: 123456789, *
- **imei** *imei_id*—Specify the subscriber IMEI. For example: 123456789012345, *
- **capture-duration** *capture_duration*—Specify the duration in seconds during which the monitor subscriber activity gets captured. The default is 300 seconds (five minutes). It's an optional parameter.
- **dump** *dump_filename*—Specify the name of the dump filename. Example: monitor subscriber-dump [filename]
- **gr-instance** *gr_instance_id*—Specify the GR instance ID. It's an optional parameter. It's a monitor subscriber for the given gr-instance only. The instance ID 1 denotes the local instance ID.
- **internal-messages** *internal_messages*—When set to yes, it enables internal messaging. By default, a disabled value It's an optional parameter.
- **list** *list_details*—Specify the details of the list. It includes the monitor subscriber list files.
- **namespace** *namespace_details*—Enable the specified namespace. By default, the namespace is set to none. It's an optional parameter.

Note A deprecated keyword in the release 2021.02.0 and replaced with nf-service as the keyword

- **nf-service** *nf_service_details*—Enable the specified NF service. By default, nf-service is set to none. It's an optional parameter. The possible values: sgw, smf, pgw

Note The nf-service keyword replaces the namespace keyword in the release 2021.02 and onwards.

- **transaction-logs** *transaction_logs*—Enable transaction logs when set to yes. By default, a disabled value. It's an optional parameter.

Note To view the transaction history logs, use the dump transaction history command. The latest transaction logs get stored in a circular queue of size 1024 transaction logs.

- For more information, see *Troubleshooting Information* > Logs.

Important The MonSub needs subscriber SUPI or IMSI in monitor subscriber command. The following are important actions:

- As a prerequisite, the SMF must print the subscriber SUPI or IMSI in the logs.
- Currently, SMF transaction logs are printing subscriber session keys which include subscriber SUPI or IMSI in the transcation logs.
- The operator can pick the subscriber SUPI or IMSI from these logs.

Configuration Example

The following is an example configuration.

```
logging level monitor-subscriber info
logging name infra.message_log.core level monitor-subscriber debug
```

Configuration Verification

To verify the configuration:

```
smf# show running-config logging
logging level monitor-subscriber info
logging name infra.message_log.core level monitor-subscriber debug
```

N40 Additional Logs and Statistics

The following are the synopsis of the additional logs and statistics for N40 to trigger session management policies towards the SMF:

- Enhances the current message-level statistics for rest-ep consisting of extra LogTag or labels, as the following:
  - DNN type

    - RAT type
    - PDN type
    - Procedure type
    - Types of error cause, when there are errors.
    - Interface failure

- When the current message-level statistics get enhanced, the redundant statistics from the SMF-Service such as Message-level statistics can be removed, as the rest-ep already consists of the same message.
- The problems and their details can be updated for all the CHF-initiated messages as the following:
    - Notify
    - For Abort
    - Re-Auth
- The transaction logging levels and the LogTag for the additional logs and statistics for N40 must be in sync. In the event of a failure scenario, the applicable LogTag for N40 must be added. They are as the following:
    - Major (Error)
    - Minor (Warning)
    - Normal (Info)
- When the logging level gets enabled with the corresponding category, the following options are available:
    - Event Trace
    - Core Dump

Configuring the N40 Additional Logs and Statistics

To configure this feature use the following configuration:

`logging name smf-service.smf-app.chf level transaction debug-level`

NOTES:

- **Warning | Error**—The logging level supported for the N40 additional logs and statistics LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
logging name smf-service.smf-app.chf level transaction warn
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name smf-service.smf-app.chf
logging name smf-service.smf-app.chf level transaction warn
```

N7 Additional Logs and Statistics

The following are the synopsis of the additional logs and statistics for N7 to trigger session management policies towards the SMF:

- It detects HTTP error codes for the PCF initiated update notification and terminate messages containing problem details.
- Enhances the current message-level statistics for rest-ep consisting of extra LogTag or labels, as the following:
  - DNN type
  - RAT type
  - PDN type
  - Procedure type
  - Types of error cause, when there are errors.
  - Interface failure
- The transaction logging levels and LogTag for the additional logs and statistics for N7 must be in sync. They are as the following:
  - Error
  - Warning
  - Info
- When the logging level gets enabled with the corresponding category, the following options are available:
  - Event Trace
  - Core Dump

Configuring the N7 Additional Logs and Statistics

To configure this feature, use the following configuration:

`logging name smf-service.smf-app.pcf level transaction debug-level`

NOTES:

- **Warning | Error**—The logging level supported for N7 additional logs and statistics LogTags.
- For more information, see *Troubleshooting Information* > Logs.

Configuration Example

The following is an example configuration.

```
logging name smf-service.smf-app.pcf level transaction warn
```

Configuration Verification

To verify the configuration:

```
[smf] smf# show running-config logging name smf-service.smf-app.pcf
logging name smf-service.smf-app.pcf level transaction warn
```