Tethering Detection

This chapter describes the Tethering Detection feature and provides detailed information on the following topics:

Feature Description

This section provides an overview of the Tethering Detection feature.

Important In this release, the Tethering Detection feature is supported only on the GGSN, HA, and P-GW.

Tethering refers to the use of a mobile smartphone as a USB dongle/modem to provide Internet connectivity to PC devices (laptops, PDAs, tablets, and so on) running on the smartphone's data plan. Typically, for smartphone users, most operators have in place an unlimited data plan, the usage of which is intended to be from the smartphone as a mobile device. However, some subscribers use the low cost / unlimited usage data plan to provide Internet connectivity to their laptops in places where normal Internet connection via broadband/WiFi may be costly, unavailable, or insecure.

The Tethering Detection feature enables detection of subscriber data traffic originating from PC devices tethered to mobile smartphones, and also provides effective reporting to enable service providers take business decisions on how to manage such usage and to bill subscribers accordingly. Tethering Detection is supported for IPv4 (TCP)and IPv6 traffic flows.

The Tethering Detection feature is enabled on a per rulebase basis. The rulebase (billing plan) assigned for APN will contain the tethering detection related configuration. ECS performs tethering detection on a per flow basis for all subscribers (for whom TAC database match succeeded) using an APN in which the feature is enabled. The extent to which the detection mechanism is executed depends on the type of flow.

ECS supports various tethering detection solutions:

- TTL-based tethering detection
- UA-based tethering detection

- OS-based tethering detection

The IP-TTL based tethering detection solution is implemented to support tethering for all IP flows - both IPv4 and IPv6. This feature is configurable at the rulebase level and will be done for all flows of all subscribers having IP-TTL configuration within the rulebase.

Important IPv6 OS-based tethering detection is supported.

IPv6 Tethering Detection is supported for OS-based signatures. If the signature format used for IPv6 OS based tethering detection needs to be modified, additional data must be collected to identify fields of the new signature. The following new TCP parameters are added to EDRs:

- IPv6 OS signature string
- Sequence number from the TCP SYN packet of a flow
- 8 bytes of control parameters that include data offset, reserved, flags, window size, checksum and urgent pointer from TCP SYN header
- TCP options, if they are present in TCP SYN header

License Requirements

Tethering Detection is a licensed Cisco feature. A separate feature license may be required. Contact your Cisco account representative for detailed information on specific licensing requirements. For information on installing and verifying licenses, refer to the *Managing License Keys* section of the *Software Management Operations* chapter in the *System Administration Guide*.

Flow Recovery Support for Tethering Detection

Feature Information

Summary Data

Status	New Functionality
Introduced-In Release	21.2
Modified-In Release(s)	Not Applicable
Applicable Product(s)	GGSN, HA, P-GW
Applicable Platform(s)	ASR 5500
Default Setting	Enabled. Flow recovery configurations are inherited.
Related CDETS ID(s)	CSCvc80454
Related Changes in This Release	Not Applicable

Related Documentation	ECS Administration Guide Statistics and Counters Reference Statistics and Counters Reference - Counter Descriptions

Revision History

Important Revision history details are not provided for features introduced before release 21.2.

Revision Details	Release	Release Date
New in this release.	21.2	April 27, 2017

Feature Changes

This feature introduces recovery support for the flows which have tethering detection enabled for them. For this, the flow recovery uses the service scheme framework to support the recovery. The recovered information helps in post SR and ICSR and continue marking tethered flows.

Previous Behavior: Flow recovery did not recover the tethering specific parameters.

New Behavior: Flow recovery now recovers tethering specific parameters in addition to existing Flow Recovery behavior.

Customer Impact: Post recovery, tethered flows might have differential charging and tethering might have visibility in the EDR.

How it Works

This section describes the working of "Recovery of tethered flag" and "Recovery of parameters for tethering the EDRs".

Recovery of Tethered Flag

This section lists the steps that occur during recovery of tethered flag.

1. The following checkpoint tethering-detection-method-types are used, by which flow is detected as tethered:
   - IP-TTL-tethered
   - OS-tethered-flag/UA-tethered
   - DNS-based-tethered
2. Check point is done as before for all flows that qualify the flow recovery trigger condition. With this enhancement, tethering flags and EDR values are also checkpointed for such flows, additionally.
3. Piggyback above tethering-recovery-info on the existing flow recovery information is done, that would be happening for trigger-type flow-create and the given trigger condition.
4. Check point is triggered when trigger condition is met (and corresponding action is flow recovery) .
5. On the first packet received, the flags are restored after recovery.

Important Post-flow-recovery UA-based-TD is not supported as per flow-recovery-framework limitations.

Recovery of Tethering EDR Parameters

This section lists the steps that occur during recovery of parameters for tethering the EDRs:

1. The following tethering specific EDR parameters are recovered for a tethered flow:
   - IP-TTL value
   - IPv4 OS signature
   - IPv6 OS signature
   - Signature-scan-status
2. Check point is done as before for all flows that qualify the flow recovery trigger condition. With this enhancement, tethering flags and EDR values are also checkpointed for such flows, additionally.
3. Piggyback above tethering-recovery-info on the existing flow recovery information is done, that would be happening for trigger-type flow-create and the given trigger condition.
4. Check point is triggered when trigger condition is met (and corresponding action is flow recovery) .
5. On the first packet received, the flags are restored after recovery.

Important Recovery support for multiple tethering detection EDR OS signatures in the same flow is not supported.

Sample Configuration

This section lists the sample configuration for "Recover IP-TTL based tethering detection" and "EDR Configuration".

Sample Configuration to Recover IP-TTL Based Tethering-Detection

```
configure
 active-charging service s1
  subscriber-base default
    priority 10 subs-class class1 bind service-scheme flow_recovery
    #priority 20 subs-class class2 bind service-scheme scheme2
  #exit

  subs-class class1
    apn = starent.com
    rulebase = plan1
    v-apn != some_virtual_apn
    multi-line-or all-lines
#exit

  rulebase plan1
```

```
    action priority 1 ruledef tethered_flow charging-action standard
    tethering-detection ip-ttl value 62
  #exit

  ruledef tethered_flow
    tethering-detection ip-ttl flow-tethered
    ip any-match = TRUE
  #exit

  service-scheme flow_recovery
    trigger flow-create
    priority 1 trigger-condition flow_rec_cond trigger-action
tether_flow_recov
  #exit

  trigger-condition flow_rec_cond
    rule-name = tethered_flow
    #any-match = TRUE
#exit

  trigger-action tether_flow_recov
    flow-recovery
  #exit
end
```

Sample EDR Configuration

```
conf
act s s1
rulebase plan1
billing-records egcdr
flow end-condition normal-end-signaling edr edr_td
end

conf
act s s1
edr-format edr_td
rule-variable flow ttl priority 1
rule-variable tcp os-signature priority 2
rule-variable tcp v6-os-signature priority 3
end
```

Monitoring and Troubleshooting

This section lists the commands available to monitor the "Flow Recovery Support for Tethering Detection" feature.

Show Commands

This section lists all the show commands available to monitor this feature.

show active-charging tethering-detection statistics

This command has been modified to display the following output:

```
Current Tethered Subscribers:                    0
    Total flows scanned:                         0
    Total Tethered flows detected:               0
    Total Tethered flows recovered:              1
    Total flows bypassed for scanning :          0

Tethering Detection Statistics (os-ua):
   TAC ID lookups:                               0
   TAC ID matches:                               0
   OS signature lookups:                         0
   OS signature matches:                         0
   IPv6 OS signature lookups:                    0
   IPv6 OS signature matches:                    0
   UA signature lookups:                         0
   UA signature matches:                         0
   Total flows scanned:                          0
   Tethered flows detected:                      0
   Non-tethered flows detected:                  0
   Tethered Uplink Packets:                      0
   Tethered Downlink Packets:                    0
   Current tethering-detected indications sent:  0
   Total tethering-detected indications sent:    0

Tethering Detection Statistics (ip-ttl):
   Total flows scanned:                          0
   Tethered flows detected:                      0
   Tethered uplink packets:                      0
   Tethered downlink packets:                    0

Tethering Detection Statistics (DNS Based):
   Total flows scanned:                          0
   Tethered flows detected:                      0
   Tethered uplink packets:                      1
   Tethered downlink packets:                    1

Change Statistics for Multiple SYN in Flow:
   Tethered to Non-Tethered:                     0
   Non-Tethered to Tethered:                     0
   Tethered to Tethered:                         0
   Non-Tethered to Non-Tethered:                 0
```

Bulk Statistics

This section lists all the bulk statistics that have been added, modified, or deprecated to support this feature.

ECS Schema

This section displays the bulk stats that have been added for total tethered recovered flows:

- ecs-td-total-recovered-flows - Total number of recovered tethered flows

IPv6 DNS-based Tethering Detection

The Tethering Detection feature is enhanced to detect DNS-based tethering for IPv6 flows. This feature can be used to detect tethering for users using stealth tethering applications such as FoxFi, EasyTether, and so on.

The DNS-based solution is implemented to address the deficiencies that other incumbent solutions had in detecting IPv6 tethered flows. Tethering detection for IPv6 traffic was achieved using either IPv6 IP-TTL based tethering detection or IPv6 OS-based tethering detection solution.

Mid-flow Tethering Detection

The Tethering Detection feature supports mid-flow tethering detection and EDR/REDR generation on tethering signature change. IP-TTL and OS-based tethering detection will analyze mid-flow SYN to generate IP-TTL/OS signature. When this feature is enabled, OS-based tethering detection keeps the OS-signature value of the last SYN packet and also generates EDR/REDR on tethering signature change. The generated EDR/REDR contains the signature of the previous SYN packet. Once the flow is classified as tethered (from first SYN or mid-flow SYN), that flow will remain tethered throughout the lifetime of the flow.

This feature provides the capability to limit the number of SYN packets that need to be analyzed on the same flow. As EDR/REDR gets generated in mid-flow due to tethering signature change, a new closure reason is introduced to track such scenarios.

The **max-syn-packet-in-flow** command in the ACS Rulebase Configuration mode is configured to limit the maximum number of SYN packets to be analyzed for tethering detection. The **flow end-condition tethering-signature-change** command in the ACS Rulebase Configuration mode is configured to control the behavior of mid-flow EDR generation due to tethering signature change.

Tethering Detection Bypass based on Interface-ID

The Tethering Detection feature is enhanced to support tethering detection bypass based on Interface ID. A 64-bit Interface ID can be configured to bypass tethering detection for flows with this interface ID matching the interface ID of the source IPv6 address.

The **bypass interface-id** keyword is added to the **tethering-detection** command in the ACS Configuration mode to bypass IP-TTL and OS based tethering detection for IPv6 packets.

- By default, tethering detection is not bypassed.
- Only one Interface ID can be configured for tethering detection bypass in the active-charging service.
- Only IPv6 OS-based and IPv6 IP-TTL-based tethering detection is bypassed. IPv4 tethering detection and UA-based tethering detection is not impacted.
- Tethering detection bypass is applicable only to IPv6 flows.
- Tethering detection bypass is applicable only to IPv6 OS database lookup and IPv6 TTL lookup. EDR generation, IPv6 OS signature generation and population of the signature in EDR will remain unaffected.

Tethering Detection Databases

The Tethering Detection database files must be populated and loaded on to the ASR chassis by the administrator. The procedure to load the database is the same for all the different databases.

Before the database(s) can be loaded for the first time, tethering detection must be enabled using the **tethering-database** CLI command in the ACS Configuration Mode.

For all the databases, only a full upgrade of a database file is supported. Incremental upgrade is not supported. If, for any particular database, the upgrade procedure fails, the system will revert back to the previous working version of that database.

OS-signatures can be collected from TCP SYN even when tethering detection is disabled in the rulebase. The os-signature will be parsed if an EDR/UDR with an os-signature variable is present in a rulebase or charging-action in the rulebase. This can be used to collect os-signatures that can then be used to build an OS database for the tethering detection feature.

TAC-db lookup for tethering detection can be enabled or disabled using the **tethering-detection tac-db** CLI command in the ACS Configuration mode. Based on the CLI configuration, TAC-db lookup needs to be performed at session setup.

Loading and Upgrading Tethering Detection Databases

This section provides an overview of loading and upgrading the databases used in tethering detection.

The database files from MUR/MURAL must be copied onto the ASR chassis to the following directory path designated for storing the database files:

/hd-raid/databases/

Any further upgrades to the database files can be done by placing the file named *new-filename* in the designated directory path. ACS auto-detects the presence of files available for upgrade daily. When a new version of a file is found, the upgrade process is triggered. The upgrade can also be forced by running the upgrade command in the CLI. On a successful upgrade this file is renamed to *filename*.

MUR/MURAL Support for Tethering Detection

The ASR chassis works in conjunction with the Mobility Unified Reporting (MUR/MURAL) application to facilitate tethering detection on the chassis.

If MUR/MURAL is not deployed, then the database file must be manually placed on the ASR chassis, in the /hd-raid/databases/ directory, and loaded into configuration using CLI command.

For more information on MUR, refer to the *MUR Online Help System* and the *Mobility Unified Reporting System Installation and Administration Guide*.

For more information on MURAL, refer to the *MURAL Online Help System* and the *Mobility Unified Reporting and Analytics System Installation and Administration Guide*.

Session Recovery Support

The following Session Recovery features are supported:

- Database recovery after SessCtrl getting killed
- Database recovery after one or more SessMgrs getting killed

Note that depending on the size of the database files and the number of SessMgrs operational in the system, it may take sometime (ranging from five seconds to five minutes) for the database to become available in all the SessMgrs post recovery/migration.

How It Works

This section describes the Tethering Detection configuration. The following examples illustrate two different implementations of the Tethering Detection configuration.

- The following type of configuration is suitable where ECS performance is critical and the operator wants to put in a flat charging plan in place for all the tethered traffic. In such a scenario, addition of a single new ruledef to the configuration suffices. Placing this ruledef at the highest priority in the rulebase will ensure all the tethered flows are charged as per the tariff plan for tethered traffic.

```
configure
     active-charging service ecs_service
          tethering-database
          ruledef tethered-traffic
               tethering-detection flow-tethered
               tcp any-match = TRUE
               exit
          ruledef ftp-pkts
               ftp any-match = TRUE
               exit
          ruledef http-pkts
               http any-match = TRUE
               exit
          ruledef tcp-pkts
               tcp any-match = TRUE
               exit
          ruledef ip-pkts
               ip any-match = TRUE
               exit
          ruledef http-port
               tcp either-port = 80
               rule-application routing
               exit
          ruledef ftp-port
               tcp either-port = 21
               rule-application routing
               exit
          charging-action premium
               content-id 1
               retransmissions-counted
               billing-action egcdr
               exit
          charging-action standard
               content-id 2
               retransmissions-counted
               billing-action egcdr
               exit
          rulebase consumer
               tethering-detection
               action priority 10 ruledef tethered-traffic charging-action premium
               action priority 20 ruledef ftp-pkts charging-action standard
               action priority 30 ruledef http-pkts charging-action standard
               action priority 40 ruledef tcp-pkts charging-action standard
               action priority 50 ruledef ip-pkts charging-action standard
               route priority 80 ruledef http-port analyzer http
               exit
          rulebase default
          end
```

- The following type of configuration is suitable when operators want to apply differentiated charging to various flows that are found to be tethered. In this case, traffic that requires different charging action or

content ID when it is tethered will be identified using two ruledefs, one with "flow-is-tethered = TRUE" option and another without this option. This configuration provides finer granularity of control but results in higher performance degradation because the rule matching tree size increases.

```
configure
     active-charging service ecs_service
          tethering-database
          ruledef ftp-pkts
               ftp any-match = TRUE
               exit
          ruledef ftp-pkts-tethered
               ftp any-match = TRUE
               tethering-detection flow-tethered
               exit
          ruledef http-pkts
               http any-match = TRUE
               exit
          ruledef http-pkts-tethered
               http any-match = TRUE
               tethering-detection flow-tethered
               exit
          ruledef tcp-pkts
               tcp any-match = TRUE
               exit
          ruledef tcp-pkts-tethered
               tcp any-match = TRUE
               tethering-detection flow-tethered
               exit
          ruledef ip-pkts
               ip any-match = TRUE
               exit
          ruledef ip-pkts-tethered
               ip any-match = TRUE
               tethering-detection flow-tethered
               exit
          ruledef http-port
               tcp either-port = 80
               rule-application routing
               exit
          ruledef ftp-port
               tcp either-port = 21
               rule-application routing
               exit
          charging-action premium-http
               content-id 10
               retransmissions-counted
               billing-action egcdr
               exit
          charging-action premium-ftp
               content-id 20
               retransmissions-counted
               billing-action egcdr
               exit
          charging-action premium
               content-id 1
               retransmissions-counted
               billing-action egcdr
               exit
          charging-action standard
               content-id 2
               retransmissions-counted
               billing-action egcdr
               exit
          rulebase consumer
```

```
    tethering-detection
   action priority 10 ruledef ftp-pkts-tethered charging-action premium-ftp

    action priority 20 ruledef ftp-pkts charging-action standard
  action priority 30 ruledef http-pkts-tethered charging-action premium-http

    action priority 40 ruledef http-pkts charging-action standard
    action priority 50 ruledef tcp-pkts-tethered charging-action premium
    action priority 60 ruledef tcp-pkts charging-action standard
    action priority 70 ruledef ip-pkts-tethered charging-action premium
    action priority 80 ruledef ip-pkts charging-action standard
    route priority 80 ruledef http-port analyzer http
    exit
rulebase default
    end
```

Configuring Tethering Detection

This section describes how to configure the Tethering Detection feature to detect subscriber flows from PC devices tethered to mobile smartphones.

Important This command is available only if the *Smartphone Tethering Detection* license is enabled. For more information please contact your Cisco account representative.

To enable and configure the Tethering Detection feature, use the following configuration:

```
configure
  active-charging service <ecs_service_name>
    tethering-database [ ipv6-os-signature ipv6_os_signature_db_file_name |
os-signature <os_signature_db_file_name> | tac <tac_db_file_name> | ua-signature
<ua_signature_db_file_name> ] +
    [ no ] tethering-detection { bypass interface-id ifid | dns-based
nat64 ipv6_network_prefix | tac-db }
    ruledef <tethering_detection_ruledef_name>
      tethering-detection [ dns-based | ip-ttl | os-ua ] {
flow-not-tethered | flow-tethered }
      exit
    rulebase <rulebase_name>
      tethering-detection { dns-based | bypass interface-id ifid | ip-ttl
 value ttl_value | max-syn-packet-in-flow max_syn_packets | os-db-only |
os-ua-db | ua-db-only }
      action priority <priority> ruledef <tethering_detection_ruledef_name>
charging-action <charging_action_name>
      ...
      end
```

Notes:

- The default filename for the IPv6 OS Signature database is **v6-os-db**. The default filename can be changed by the user only during boot up. Once the system is up and running, the database file name cannot be modified. This is true for all tethering related database files.

- In release 20.2, the **bypass interface-id** *ifid* keyword is added in the ACS Rulebase Configuration mode to bypass IP-TTL and OS based tethering detection for IPv6 packets. *ifid* specifies the Interface ID from an IPv6 address, that is a 64-bit unsigned integer.

  When configured, all the IPv6 flows having this interface ID in the source IP address will bypass IP-TTL and OS based tethering detection.

- In release 20.2, the **max-syn-packet-in-flow** *max_syn_packets* keyword is added in the ACS Rulebase Configuration mode to determine the number of SYN packets applicable for tethering detection in a flow. *max_syn_packets* must be an integer ranging from 1 to 3.

  Default number of SYN packets is 1. This means that only the first SYN packet in flow will be analyzed for IP-TTL/OS signature generation and tethering detection. All other mid flow SYN packets will be ignored for IP-TTL/OS signature generation and tethering detection.

- In release 21, the **dns-based nat64** *ipv6_network_prefix* keyword is added in the ACS Configuration mode to configure DNS-based lookup for tethering detection. *ipv6_network_prefix* must be an IPv6 colon-separated-hexadecimal notation with subnet mask bit. IPv6 also supports :: notation.

  The configured NAT64 prefixes are used to identify the IPv6 flows that will be considered for DNS-based tethering detection.

- In release 21, the **dns-based** keyword is added in the ACS Rulebase Configuration mode to perform tethering detection based on DNS pattern. When DNS-based tethering detection is configured, this feature is enabled for all subscribers using this billing plan.

- In release 21, the **dns-based** keyword is added in the ACS Ruledef Configuration mode to match traffic identified as tethered or non-tethered by the DNS-based detection solution.

Enabling TAC Database Lookup

To enable TAC-db lookup for tethering detection in the ACS configuration mode, use the following configuration.

```
configure
   active-charging service service_name
      [ no ] tethering-detection { tac-db }
      end
```

Notes:

- **no tethering-detection tac-db**: Skips TAC-db lookup for tethering detection.
- Enabling TAC-db lookup for tethering detection is the default behavior.

Verifying your Configuration

To verify your configuration, in the Exec Mode, enter the following command:

```
show active-charging subscribers full all
```

The **Tethering Detection Enabled** field displays whether tethering detection for a subscriber is enabled or not.

Enabling DNS Caching

Use the following configuration to allow caching from DNS flows when the DNS-based tethering detection is enabled.

```
configure
   active-charging service service_name
      charging-action charging_action_name
         flow tethering-detection dns-based host-table caching
         { default | no } flow tethering-detection
         end
```

Upgrading Tethering Detection Databases

To upgrade the Tethering Detection feature databases, in the Exec mode, use the following CLI command:

```
upgrade tethering-detection database { all | ipv6-os-signature |
os-signature | tac | ua-signature } [ -noconfirm ]
```

Notes:

- To load and upgrade the databases used in detecting tethering, the database files must be copied from MUR/MURAL onto the ASR chassis to the designated directory path for storing the database files:

  `/mnt/hd-raid/data/databases/`

  Any further upgrades to the database files can be done by placing the file named *new-filename* in the designated directory path. ECS auto-detects the presence of files available for upgrade daily. When a new version of a file is found, the upgrade process is triggered. The upgrade can also be forced by running the upgrade command in the CLI. On a successful upgrade this file is renamed to *filename*.

Monitoring and Troubleshooting the Tethering Detection feature

This section provides information regarding show commands and/or their outputs in support of this feature.

Tethering Detection Show Command(s) and/or Outputs

This section provides information regarding show commands and/or their outputs in support of the Tethering Detection feature.

Bulk Statistics

Bulk statistics reporting for the Tethering Detection feature is supported.

The following bulk statistics are available in the ECS schema:

- ecs-td-tac-id-lookups
- ecs-td-tac-id-matches
- ecs-td-os-signature-lookups
- ecs-td-os-signature-matches
- ecs-td-ua-signature-lookups

- ecs-td-ua-signature-matches
- ecs-td-v6-os-signature-lookups
- ecs-td-v6-os-signature-matches
- ecs-td-total-flows-scanned
- ecs-td-tethered-flows-detected
- ecs-td-non-tethered-flows-detected
- ecs-td-osua-total-flows-scanned
- cs-td-osua-tethered-flows-detected
- ecs-td-osua-non-tethered-flows-detected
- ecs-td-ipttl-total-flows-scanned
- ecs-td-ipttl-tethered-flows-detected
- ecs-td-ipttl-non-tethered-flows-detected
- ecs-td-current-tethered-subscribers
- ecs-td-tethered-uplink-packets
- ecs-td-tethered-downlink-packets
- ecs-td-ipttl-tethered-uplink-packets
- ecs-td-ipttl-tethered-downlink-packets
- ecs-td-tether-to-tether-signature-change-in-flow
- ecs-td-tether-to-non-tether-signature-change-in-flow
- ecs-td-non-tether-to-tether-signature-change-in-flow
- ecs-td-non-tether-to-non-tether-signature-change-in-flow

For more information on these bulk statistics, see the *ECS Schema Statistics* chapter of the *Statistics and Counters Reference*.

SNMP Traps

SNMP traps indicate the load/upgrade status for the TAC, UA, OS and IPv6 OS signature tethering databases.

The following SNMP traps are added:

- starTetheringOSDatabaseUpgradeFailureStatus - OS database upgrade failure status
- starTetheringOSDatabaseUpgradeSuccessStatus - OS database upgrade success status
- starTetheringTACDatabaseUpgradeFailureStatus - TAC database upgrade failure status
- TetheringTACDatabaseUpgradeSuccessStatus - TAC database upgrade success status
- starTetheringUADatabaseUpgradeFailureStatus - UA database upgrade failure status
- starTetheringUADatabaseUpgradeSuccessStatus - UA database upgrade success status
- starTetheringV6OSDatabaseUpgradeFailureStatus - IPv6 OS database upgrade failure status
- starTetheringV6OSDatabaseUpgradeSuccessStatus - IPv6 OS database upgrade success status

For more information regarding SNMP MIB objects, refer to the *SNMP MIB Reference*.