MobileNext Broadband Gateway

Monitoring and Troubleshooting the Mobile Environment

Published: 2012-04-11

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

MobileNext Broadband Gateway Monitoring and Troubleshooting the Mobile Environment

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- MX240 Routers
- MX960 Routers
- MX480 Routers

Documentation Conventions

Table 1 on page viii defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page viii defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the **configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`
Italic text like this	• Introduces important new terms. • Identifies book names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS System Basics Configuration Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; interface names; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Enclose optional keywords or variables.	**stub** <**default-metric** *metric*>**;**

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** (***string1*** **|** ***string2*** **|** ***string3***)
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Enclose a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

Indention and braces ({ })	Identify a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
J-Web GUI Conventions		
Bold text like this	Represents J-Web graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of J-Web selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to **techpubs-comments@juniper.net**, or fill out the documentation feedback form at https://www.juniper.net/cgi-bin/docbugreport/ . If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable)

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract,

or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf .
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/ .
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: https://www.juniper.net/alerts/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/ .
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see http://www.juniper.net/support/requesting-support.html .

PART 1

Overview

CHAPTER 1

Monitoring Overview

Monitoring the Mobile Environment - Key Performance Indicators

This topic describes the most common key performance indicators that you can use to determine the health of the Junos Mobility environment.

These key performance indicators include:

- GTP signaling statistics
- Session status indicators
- CPU utilization indicators
- Memory utilization indicators
- Monitored resource usage indicators (Address pools, system/APN bandwidth usage, Packet Forwarding Engine load, and so on)
- AAA authentication or accounting metrics
- Charging gateway status, congestion indicators with round trip time calculations
- Data path measurements
- Per server statistics for AAA, GTP, and CG
- Data rate measurements per configured interval

Related Documentation

- Monitoring Data Rate Statistics on page 11
- Monitoring Data Path Measurements on page 10
- Monitoring Session Status on page 6
- Monitoring GTP Signaling on page 6

Call Trace Overview

The MobileNext Broadband Gateway features many ways to monitor performance while the broadband gateways, either Gateway GPRS Support Nodes (GGSNs), Packet Data Network Gateways (P-GWs), or Serving Gateways (S-GWs), configured are in operational mode. In addition to the usual logging and alarm capabilities, the broadband gateways offer a call trace capability that allows operators with the required privileges to trace the progress of a call through the gateway in several ways, for example, by Mobile Station ISDN (MS-ISDN) or International Mobile Subscriber Identifier (IMSI).

You can initiate a call trace based on several criteria:

- Access Point Name (APN) (P-GW only)
- FPC slot
- PIC slot
- IMSI
- MS-ISDN
- Next call (up to fifty)

You can perform several actions to manage the call trace monitor operation:

- Start the subscriber events trace
- Show the subscriber events trace information
- Stop the subscriber events trace
- Clear the subscriber events trace

Call trace offers another tool for broadband gateway performance monitoring and troubleshooting.

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Example: Monitoring a P-GW with Call Trace on page 37
- Example: Monitoring an S-GW with Call Trace on page 40

CHAPTER 2

Monitors

Monitoring Resources

To avoid overload conditions, monitor the following resources:

- Detailed control plane snapshot of number of bearers per state
- Number of bearers waiting for authentication, address allocation, data path setup, and so on
- CPU of each session PIC
- Memory consumed on each session PIC
- Maximum bearer limit
- Anchor Packet Forwarding Engine load
- Individual session PIC average load
- System data path bandwidth for assured quality of service
- Queue depths (AAA, charging, GTP input, and so on)
- External interfaces like RADIUS Charging Gateway by tracking success/fail, monitoring round trip time, and so on
- Internal resource usage for local pool addresses available and so on

Monitoring GTP Signaling

To monitor GTP signaling, you can examine the messages and byte counts on Gn, S5, Gp, and S8 interfaces (statistics per APN, QCI per ARP, per GTP version, global).

You can also examine:

- Per peer history
- Per GTP cause code statistics for granular measurement of the number of failures
- Separate session establishments attempts/success counts
- Separate statistics for IPv4, IPv6, and dual address stack sessions

The following examples show how you can monitor GTP on the P-GW from the CLI.

1. To see the state of services PICs and PFEs, enter this command:

   user@host> **show unified-edge ggsn-pgw resource-manager clients**

   ```
   Client    State      Redundancy
                           Role
   pfe-0/2/0 In-Service Primary
   pfe-0/0/0 In-Service Primary
   ms-4/0/0  In-Service Primary
   ```

2. To see the resource management filters for GTP packet steering, enter the command:

   user@host> **show unified-edge rmps filters**

3. To see a summary of subscribers on the gateway, enter the command:

   user@host> **show unified-edge ggsn-pgw status detail**

4. To see subscriber details, enter the command:

   user@host> **show unified-edge ggsn-pgw subscribers extensive**

5. To show all GTP statistics (including messages sent and received, and cause codes sent and received), enter the command:

   user@host> **show unified-edge ggsn-pgw gtp statistics detail**

6. To see all the GTP peers, enter the command:

   user@host> **show unified-edge ggsn-pgw gtp peer detail**

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

Monitoring Session Status

Current session/ bearer counts can be monitored at various levels. For example, per APN, per QCI/ARP, global, per RAT type, per APN, or per QCI.

A useful command to show these types of statistics is:

```
user@host> show unified-edge ggsn-pgw qos statistics ?

Possible completions:
 <[Enter]>        Execute this command
 apn              APN name
 arp              GTPv2 ARP Value (1..15)
 gateway          Show subscriber for a gateway
 gtpv1-arp        GTPv1 ARP Value (1..3)
 qci              Show QCI statistics information (1..9)
 traffic-class    Show statistics for a traffic-class level
 traffic-handling-priority  Traffic handling priority (1..3)
 |                Pipe through a command
{backup}[edit]
user@host>
```

Use this command to examine session status indicators at the APN, gateway, and other levels.

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

Monitoring CPU Indicators

Monitoring CPU utilization relies on gathering data from session PICs.

To see status indicators for all PICs, enter:

```
user@host> show unified-edge ggsn-pgw status detail
```

To see status indicators for the gateway, enter:

```
user@host> show unified-edge ggsn-pgw status detail

Mobile gateway status of fpc slot: 0  pic slot: 0
  State          :          Backup
  Active Subscribers  :        99652
  Active Sessions    :        99652
  Active Bearers     :        99652
  CPU Load (%)       :        0
  Memory Load (%)    :        29

  Mobile gateway status of fpc slot: 0  pic slot: 1
  State          :          Active
  Active Subscribers  :        99652
  Active Sessions    :        99652
  Active Bearers     :        99652
  CPU Load (%)       :        3
  Memory Load (%)    :        97

  Mobile gateway status of fpc slot: 2  pic slot: 0
  Active Subscribers  :        0
  Active Sessions    :        0
  Active Bearers     :        0
  CPU Load (%)       :        0
```

```
Memory Load (%)    :         25

Mobile gateway status of fpc slot: 2  pic slot: 1
Active Subscribers  :          0
Active Sessions     :          0
Active Bearers      :          0
CPU Load (%)        :          0
Memory Load (%)     :         25
```

To see status indicators for an individual PIC (in the example shown, fpc-slot 2 pic-slot 0), enter:

```
user@host> show unified-edge ggsn-pgw status fpc-slot 2 pic-slot 0
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5
- Monitoring Memory Indicators on page 8

Monitoring Memory Indicators

You can monitor system memory by gathering data from session PICs just as you do for CPU usage.

To see memory indicators for all PICs, enter:

```
user@host> show unified-edge ggsn-pgw status detail
```

To see memory indicators for an individual PIC (in the example shown, fpc-slot 2 pic-slot 0), enter:

```
user@host> show unified-edge ggsn-pgw status fpc-slot 2 pic-slot 0
```

Additionally, users with vty command privileges can check system load as well. This is an average of CPU and memory load and displays as “current system load.” This command is:

```
user@host> show mcos gw-resourcetbl
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5
- Monitoring CPU Indicators on page 7

Monitoring Charging Gateways

Charging gateways can be monitored by checking status, pending CDR counts, and per transport profile.

The specific statistics you can gather per charging gateway are:

- Status (alive or dead)
- Number of echo requests: transmitted, received, and timeouts
- Number of echo responses: transmitted and received
- Number of version unsupported packets: transmitted and received
- Number of node alive requests: transmitted and received
- Number of node alive responses: transmitted and received
- Number of redirection requests: received
- Number of redirection responses: transmitted
- Number of data record transfer requests: transmitted and timeouts
- Number of data record transfer success responses: received
- Total round trip time of previous DRT (avg, max, min)

The following commands are examples of charging gateway statistics:

```
user@host> show unified-edge ggsn-pgw charging path stat
Charging Path Status Peer-Addr Peer-Name Local-Address Status Echo
  1.1.1.1 cg1 10.10.10.10 Down Enabled

  Charging Path Status
  Peer-Addr      Peer-Name     Local-Address    Status       Echo
  1.1.1.1        cg1           10.10.10.10      Down         Enabled


user@host> show unified-edge ggsn-pgw charging path statistics

  Charging Path Statistics

    CGF Address                 : 1.1.1.1      CGF Server Name        : cg1
    Echo Requests               Rx: 0          Echo Responses         Tx: 0
    Echo Responses              Rx: 0          Echo Requests          Tx: 6711
    Node-Alive Requests         Rx: 0          Node-Alive Responses   Tx: 0
    Version Not Supported       Rx: 0          Version Not Supported  Tx: 0
    Echo Requests timed out     : 6710         Echo Interval          : 70
    Down Detection Interval     : 10           Reconnect Time Interval : 10
    Destination Port            : 3386         Pending Queue Size     : 0
    Path Manager FPC Slot       : 1            Path Manager PIC Slot  : 0
    T3 Response Time Interval   : 5            Path Manager Port      : 30241
    Source Interface Valid      : Yes          GTPP Header Type       : long
    N3 Requests                 : 3            Local Address          : 10.10.10.10
   GTPP Version                 : V0           Transport Protocol     : UDP

user@host> show unified-edge ggsn-pgw charging transfer status

    Charging Transfer Status
    Transport-Profile : tp1
    Total UnAck CDR's       : 0
    Total Buffered CDR's    : 0

user@host> show unified-edge ggsn-pgw charging path statistics
```

```
Charging Path Statistics

CGF Address          : 1.1.1.1      CGF Server Name       : cg1
Echo Requests        Rx: 0          Echo Responses        Tx: 0
Echo Responses       Rx: 0          Echo Requests         Tx: 6711
Node-Alive Requests  Rx: 0          Node-Alive Responses  Tx: 0
Version Not Supported Rx: 0         Version Not Supported Tx: 0
Echo Requests timed out : 6710      Echo Interval         : 70
Down Detection Interval : 10        Reconnect Time Interval : 10
Destination Port     : 3386         Pending Queue Size    : 0
Path Manager FPC Slot  : 1          Path Manager PIC Slot  : 0
T3 Response Time Interval : 5       Path Manager Port     : 30241
Source Interface Valid  : Yes       GTPP Header Type      : long
N3 Requests          : 3            Local Address         : 10.10.10.10
GTPP Version         : V0           Transport Protocol    : UDP
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

Monitoring Data Path Measurements

Data path measurements include:

- Data path Gn statistics, including the number of incoming/outgoing GTP data packets/octets on the Gn interface
- Number of discarded GTP data packets
- Data path charging statistics, including per rating-group (bearer) up and down packets/bytes
- Data path Gi/IP measurements (does not include drops on the Gi Packet Forwarding Engine)
- Incoming and outgoing packets/octets on the Gi interface
- Discarded packets
- Data path debug and miscellaneous statistics (includes number of in-progress sessions, deleting sessions, source address violations, per APN ACL violations, and so on).
- Accurate per subscriber packet/byte statistics
- Per Traffic Class packet and byte counts statistics (also per APN, global)
- IP measurements

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

Monitoring Call Rate Statistics

The following metrics are available in real time to monitor performance of the gateway call-rate indicators:

- Real-time measure of number of calls set up in the previous configurable interval
- Real-time measurement of session deactivations displayed per configurable interval
- Total data packets processed by the gateway in the past configured interval
- Total bytes of traffic handled by the gateway in the past interval

Monitoring Data Rate Statistics

To monitor data rate statistics, enter:

```
user@host> show unified-edge ggsn-pgw statistics

  Control plane statistics:
    Session establishment attempts:         1
    Successful session establishments:      1
    MS/peer initiated session deactivations:  2
    Successful MS/peer initiated deactivations: 2
    Gateway initiated session deactivations:  0
    Successful gateway initiated deactivations: 0
  Data plane GTP statistics (Gn/S5/S8):
    Input  packets:       0
    Input  bytes:         0
    Output  packets:      7751
    Output  bytes:        7251652
    Discarded packets:      0
  Data plane GTP statistics (Gi):
    Input  packets:       7751
    Input  bytes:         7251652
    Output  packets:      0
    Output  bytes:        0
    Discarded packets:      0
```

The following commands can be used for data plane statistics. There are two sets of statistics (one for the Gn interface and another for the Gi interface). The commands can be used either at the APN or the gateway level.

1. To see the gateway data plane statistics, use this command:

```
user@host> show unified-edge ggsn-pgw statistics gateway gateway-name

  Control plane statistics:
    Session establishment attempts:         0
    Successful session establishments:      0
    MS/peer initiated session deactivations:  0
    Successful MS/peer initiated deactivations: 0
    Gateway initiated session deactivations:  0
    Successful gateway initiated deactivations: 0
  Data plane GTP statistics (Gn/S5/S8):
```

```
    Input  packets:      0
    Input  bytes:        0
    Output  packets:     0
    Output  bytes:       0
    Discarded packets:     0
  Data plane GTP statistics (Gi):
    Input  packets:      0
    Input  bytes:        0
    Output  packets:     0
    Output  bytes:       0
    Discarded packets:     0
```

2. To see the APN data plane statistics, use this command:

```
user@host> show unified-edge ggsn-pgw apn statistics apn-name apn-name

  Control plane APN statistics:
    Session establishment attempts:        0
    Successful session establishments:      0
    MS/peer initiated session deactivations:  0
    Successful MS/peer initiated deactivations: 0
    Gateway initiated session deactivations:  0
    Successful gateway initiated deactivations: 0
    MS initiated modification attempts:     0
    Successful MS initiated modifications:    0
    PGW/GGSN initiated modification attempts:  0
    Successful PGW/GGSN initiated modifications:0
    User authentication statistics:
      Authentication failures:         0
      Attempted authentications:        0
      Successful authentications:       0
    Address allocation statistics:
      dynamic IP allocation attempts:     0
      dynamic IP allocation success:      0
    Charging statistics:
      Number of CDRs allocated:         0
      Number of partial CDRs allocated:    0
      Number of CDRs closed:           0
      Number of containers closed       0
    Session Establishments Failed (by GTP cause):
      Others          0
      Service unavailable: 0
      System failure:    0
      No resources:     0
      No address:      0
      Service denied:    0
      Authentication Fail: 0
      APN access denied:  0
  Miscellaneous Packet statistics:
    IPv6 Router Solicitations received:     0
    IPv6 Router Advertisement transmitted:    0
  Data plane GTP statistics (Gn/S5/S8):
    Input  packets:      0
    Input  bytes:        0
    Output  packets:     0
    Output  bytes:       0
```

```
    Discarded packets:       0
  Data plane GTP statistics (Gi):
    Input   packets:       0
    Input   bytes:         0
    Output  packets:       0
    Output  bytes:         0
    Discarded packets:     0
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

CHAPTER 3

Troubleshooting Overview

Troubleshooting Mobility

This chapter is about taking the proper action to restore network health when performance or processes are not behaving as expected.

The topics discussed in this chapter include:

- Troubleshooting overload conditions
- Troubleshooting GTP session
- Troubleshooting call admission control
- Troubleshooting AAA

Related Documentation

- Troubleshooting Overload Conditions in the Mobile Network on page 61
- Troubleshooting GTP on page 62
- Troubleshooting Alarms, Logs, and Traps on page 65
- Troubleshooting Admission Control on page 67
- Monitoring AAA Metrics on page 68

PART 2

Administration

CHAPTER 4

Monitoring Examples

Tracing Control Packets

Requirements

This example uses the following hardware and software components:

- An operational MX chassis
- Junos OS Mobility package

Tracing Control Packets

This section shows how to trace Gi to Gn control packets.

To efficiently monitor the data path, perform the following checks:

- Verify that the Gn interface (IFL) is receiving packets.

```
user@host> show jnh if statistics

    IFL Name        Index   In(Packets/Bytes)   Out(Packets/Bytes)
---------------- ---------- -------------------- --------------------
```

Verify that the packets are hitting the filter.

```
user@host>show filter
```

```
Program Filters:
---------------
  Index    Dir    Cnt   Text    Bss  Name
-------- ------ ------ ------ ------ --------

Term Filters:
------------
  Index   Semantic   Name
-------- ---------- ------
     1 Classic   __default_bpdu_filter__
 17000 Classic   __default_arp_policer__
 57008 Classic   __cfm_filter_shared_lc__
 65280 Classic   __auto_policer_template__
 65281 Classic   __auto_policer_template_1__
 65282 Classic   __auto_policer_template_2__
 65283 Classic   __auto_policer_template_3__
 65284 Classic   __auto_policer_template_4__
 65285 Classic   __auto_policer_template_5__
 65286 Classic   __auto_policer_template_6__
 65287 Classic   __auto_policer_template_7__
 65288 Classic   __auto_policer_template_8__
46137345 Classic   HOSTBOUND_IPv4_FILTER
46137346 Classic   HOSTBOUND_IPv6_FILTER
67108864 Classic   __mobile_gw_impl_filter__    <<<<<<<<<<
```

Display counters for index 67108864.

```
user@host> show filter index 67108864 counters

Filter Counters/Policers:
  Index         Packets          Bytes  Name
-------- -------------------- -------------------- --------
67108864              5          1025 __gtpc_pkt_count
67108864              5           500 __gtpu_pkt_count
67108864              0             0 __mgw_ip_frags_count
67108864              0             0 __sp-1-0_GTP_pkt_count
67108864              0             0 __sp-1-0_LIBAAA_pkt_count
67108864              0             0 __sp-1-0_LIBCHRG_pkt_count
```

Check the group TEID route table.

```
user@host> show route gtp-c

1/8                    Service  653
1.0/13                 Service  653
1.0.0.0.0/40             Service  653
```

NOTE: The *1* at the beginning here is the GTP-C route-type. If the *teid=0* route (1.0.0.0.0/40) is missing, verify that rmpsd on the Routing Engine installed those routes or verify that it is running.

```
user@host> show filter nexthops

                Name Protocol       Type Option Refcount  NH ID
```

```
------------------------------ -------- ------------ ------ -------- -----
  ms-1/0/0.16000:mgw:SPFE-AAA   IPv4    service  0x01     0   650
 ms-1/0/0.16000:mgw:SPFE-CHRG   IPv4    service  0x01     0   648
  ms-1/0/0.16000:mgw:SPFE-SMA   GTP-U   service  0x01     0   653
 ms-1/0/0.16000:mgw:SPFE-UPIC   GTP-U   service  0x01     0   656
```

Display details on service 653.

```
user@host> show nhdb id 653 extensive

   ID    Type    Interface   Next Hop Addr   Protocol    Encap    MTU    Flags  PFE internal
  Flags
  ----- -------- ------------ --------------- ---------- ------------ ---- ----------
  --------------------
   653  Service  -           -               GTP-U        -    0  0x00000000 0x00000000
  Target NH: 654
  PFE#0, Target Addr = 0x1fcbc1
      SvcDesc   = 0x1fcbff
  PFE#1, Target Addr = 0x1fcbfe
      SvcDesc   = 0x1fcbfd
```

Verify the nexthop target 654.

```
user@host> show nhdb id 654 extensive

   ID   Type    Interface   Next Hop Addr   Protocol    Encap    MTU    Flags  PFE internal
  Flags
```

Check whether the NH-id point to the correct ms-ifl.

```
user@host> show mobile-edge halp ucode-nhs

  Nexthop ID                Purpose
  ------------------------------------------------

  4194306                   GTPv0 parsing ucode
  4194307                   GTP-C v1/v2 parsing ucode
  4194308                   GTP-U swap ports ucode
  4194309                   DHCP parsing ucode
  4194310                   GTP-C table NH
  4194311                   GTP-U table NH
```

Check to see whether packets are discarded or punted the to host.

```
user@host> show jnh 0 exceptions

  Ucode Internal ---------------------- mcast stack overflow
  ...

  Packet Exceptions
  ----------------------
  bad ipv4 hdr checksum          DISC( 2)
  non-IPv4 layer3 tunnel         DISC( 4)       0       0
  GRE unsupported flags          DISC( 5)       0       0
```

```
tunnel pkt too short          DISC( 6)        0       0
bad IPv6 options pkt          DISC( 9)        0       0
bad IP hdr                    DISC(11)        0       0
bad IP pkt len                DISC(12)        0       0
L4 len too short              DISC(13)        0       0
invalid TCP fragment          DISC(14)        0       0
mtu exceeded                  DISC(21)        0       0
frag needed but DF set        DISC(22)        0       0
ttl expired                   PUNT( 1)        0       0
IP options                    PUNT( 2)        0       0
control pkt punt via ucode    PUNT( 4)        0       0
frame format error            DISC( 0)
tunnel hdr needs reassembly   PUNT( 8)        0       0
GRE key mismatch              DISC(76)        0       0
my-mac check failed           DISC(28)
frame relay type unsupported  DISC(38)        0       0
IGMP snooping control packet  PUNT(12)        0       0
bad CLNP hdr                  DISC(43)        0       0
bad CLNP hdr checksum         DISC(44)        0       0
incorrect length in GTP header DISC(45)       0       0
GTP header errors             DISC(46)        0       0
Bearer using different IP address DISC(47)    0       0
expecting sequence number     DISC(48)        0       0
sequence number isnt correct  DISC(49)        0       0
SR is marked for traffic discard DISC(50)     0       0

Firewall
----------------------
mac firewall                  DISC(78)
firewall discard              DISC(67)        0       0
tcam miss                     DISC(16)        0       0
firewall reject               PUNT(36)        0       0
firewall send to host         PUNT(53)        0       0

Routing
----------------------
discard route                 DISC(66)        0       0
hold route                    DISC(70)        0       0
mcast rpf mismatch            DISC( 8)        0       0
resolve route                 PUNT(33)        0       0
control pkt punt via nh       PUNT(34)        0       0
host route                    PUNT(32)     2313    92940
ICMP redirect                 PUNT( 3)        0       0
mcast host copy               PUNT( 6)        0       0
reject route                  PUNT(40)        0       0

Misc
----------------------
debug                         DISC(65)        0       0
services pkt internal test    PUNT(38)        0       0
directed bcast                DISC(89)        0       0
virtual-chassis pkt(hi)       PUNT(54)        0       0
virtual-chassis pkt(lo)       PUNT(55)        0       0
virtual-chassis error         DISC(42)        0       0
ME-subcriber policing out of spec packet dropsDISC(52)        0       0
```

To display non-zero counters, enter:

```
host@user> show jnh 0 exceptions terse

Reason                Type      Packets    Bytes
=============================================================

Routing
----------------------
host route            PUNT(32)     2393    96140
```

Another example is: a v2 call comes in with QCI 5 and gets mapped to FC af5. On that queue, you can see the PPS for the Gn-facing interface and the Gi-facing interface

```
user@host> show interfaces queue ge-1/2/5 forwarding-class af5
```

Where 1/2/5 is the Gn-facing interface. Then enter:

```
user@host> show interfaces queue ge-1/2/1 forwarding-class af5
```

Where 1/2/1 is the Gi-facing interface.

Configuration

- Tracing Packets on page 23

Tracing Packets

Results This example illustrated the steps you can take to trace control packets.

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

How to Trace Data Packets from Gn to Gi Interfaces

- Requirements on page 23
- Tracing Data Packets on page 23
- Configuration on page 23

Requirements

This example uses the following hardware and software components:

- An operational MX chassis
- Junos OS Mobility package

Tracing Data Packets

This section shows how to trace Gn to Gi (GTP-U) data packets.

Configuration

- Setting up Data Packet Tracing on page 24

Setting up Data Packet Tracing

Step-by-Step Procedure

The following procedure shows how to trace GTP-U (Gn to Gi) data packets.

1. Verify that the Gn/S5 interface is receiving packets.

```
user@host> show jnh if statistics

    IFL Name          Index       In(Packets/Bytes)       Out(Packets/Bytes)
...
```

2. Verify that the filter is seeing GTP-U packets by finding the index of the implicit RTT filter used by mobility applications.

```
user@host> show filter UE address prefix

Program Filters:
---------------
  Index    Dir    Cnt    Text    Bss  Name
-------- ------ ------ ------ ------ --------

Term Filters:
------------
  Index   Semantic   Name
-------- ---------- ------
     2 Classic   __default_bpdu_filter__
 17000 Classic   __default_arp_policer__
 57008 Classic   __cfm_filter_shared_lc__
 65280 Classic   __auto_policer_template__
 65281 Classic   __auto_policer_template_1__
 65282 Classic   __auto_policer_template_2__
 65283 Classic   __auto_policer_template_3__
 65284 Classic   __auto_policer_template_4__
 65285 Classic   __auto_policer_template_5__
 65286 Classic   __auto_policer_template_6__
 65287 Classic   __auto_policer_template_7__
 65288 Classic   __auto_policer_template_8__
46137345 Classic   HOSTBOUND_IPv4_FILTER
46137346 Classic   HOSTBOUND_IPv6_FILTER
46137347 Classic   __me_uplink_exception_filter_ipv4__
46137349 Classic   __me_uplink_exception_filter_ipv6__
67108864 Classic   __mobile_gw_impl_filter__        <----
```

3. Verify that the filter has the correct GGSN IP address (172.23.9.100 in the following output) in the filter.

```
user@host> show filter index 67108864 program

Filter index = 67108864
Optimization flag: 0xf7
Filter notify host id = 0
Filter properties: None
Filter state = CONSISTENT
term IP-Fragments
term priority 0
```

```
    is-fragment
      value & 0x3fff != 0x0000
      false branch to match protocol in rule GTP-U
    destination-address
      172.23.9.100/32 <----
      false branch to match protocol in rule GTP-U

    then
      action next-hop, type (nh-id)
          4194308
      count __mgw_ip_frags_count
  term GTP-U
  term priority 0
    protocol
       17
      false branch to match action in rule default-term
    port
       2152
      false branch to match port in rule GTP-C
    destination-address
      172.23.9.100/32 <----
      false branch to match port in rule GTP-C

    then
      action next-hop, type (nh-id)
          4194308
      count __gtpu_pkt_count
  term GTP-C--
  term priority 0
    port
       2123
      false branch to match port in rule sp-1/0_LIBAAA_udp_start_10000
    destination-address
      172.23.9.100/32
      false branch to match port in rule sp-1/0_LIBAAA_udp_start_10000

    then
      action next-hop, type (nh-id)
          4194307
      count __gtpc_pkt_count
```

4. Verify that the implicit RTT firewall filter counter for GTP-U is incrementing.

```
user@host> show filter index 67108864 counters

Filter Counters/Policers:
  Index         Packets          Bytes  Name
-------- -------------------- -------------------- --------
67108864               2          452  __gtpc_pkt_count
67108864               2          584  __gtpu_pkt_count <----
67108864               0            0  __mgw_ip_frags_count
67108864               0            0  __sp-1-0_GTP_pkt_count
67108864               0            0  __sp-1-0_LIBAAA_pkt_count
67108864               0            0  __sp-1-0_LIBCHRG_pkt_count
```

5. Verify that the GTP-U ingress ucode NH is created and is not marked as *Discard*.

```
user@host> show mobile-edge halp ucode-nhs

  Nexthop ID                Purpose
  ------------------------------------------------

  4194306                   GTPv0 parsing ucode
  4194307                   GTP-C v1/v2 parsing ucode
  4194308                   GTP-U ingress ucode
  4194309                   DHCP parsing ucode
  4194310                   GTP-C table NH
  4194311                   GTP-U table NH
  4194312                   GTP-U restore packet context ucode
  4194313                   IP frag load balancing ucode
```

```
host@user> show nhdb id 4194308 extensive

  ID   Type   Interface  Next Hop Addr   Protocol   Encap   MTU   Flags PFE internal
   Flags
  4194308  Unicast -        -           GTP-U       -   0  0x00000000 0x00008004

  Flags:          0x00000000
  PFE internal flags: 0x00008004

  Dram Bytes    : 268
  PreComputed MTU: 0
  Flags         : 0x0
  Parent NHID   :0


  PFE:0

  Encap-ptr chain:
  ----------------

  Dram Bytes: 268
```

6. Verify that the GTP-U route table is set up correctly on the Gn ingress Packet Forwarding Engine.

```
host@user> show route gtp-u

  default                   Service  653
  0.0.0.0                   Service  647
  0.32/16                   Unicast  666 mif.16000
```

NOTE: Data TEID route should be present in the gtp-u table (0.32/16 in this case, which is **teid starting with 0x02**). Since any Packet Forwarding Engine can be ingress Packet Forwarding Engine, the same GTP-U route table is present on all Packet Forwarding Engines. NH-id 666 in the route corresponds to the anchor Packet Forwarding Engine.

7. To find the anchor Packet Forwarding Engine, execute the following command. The L2 interface identifies the anchor Packet Forwarding Engine. In the output below, fpc 0 pic 0 is the anchor Packet Forwarding Engine.

```
host@user> show nhdb id 666 extensive

   ID  Type  Interface  Next Hop Addr  Protocol  Encap  MTU  Flags PFE internal
   Flags
    666  Unicast  mif.16000  default          IPv4  Unspecified  0  0x10000000
   0x00000000

   Flags:          0x10000000
   PFE internal flags: 0x00000000
    L2-Interface: pfe-0/0/0.16383 (81) <------- ANCHOR PFE

   Dram Bytes    : 268
   PreComputed MTU: 0
   Flags       : 0x10000000
   Parent NHID   :0
        Feature List: NH
          [pfe-0]: 0xce811db000200005;
         f_mask:0x00400000; c_mask:0x80000000; f_num:11; c_num:1,  inst:0
        Idx#9     ucast:
          [pfe-0]: 0xce811db000200005

   <.....SNIP.....>
```

8. Verify that the subscriber is installed in the anchor Packet Forwarding Engine (LU id = 0 here).

```
host@user> show vbf hw 0 subscriber-table uplink

+----------+----------+--------+--------+--------+-------+--------+----------+----------+
| RINDEX   | TEID     | VRF ID | TFT ID | CONFIG | FLAGS | CHRGID | CHRGADDR |
IPADDR   |
+----------+----------+--------+--------+--------+-------+--------+----------+----------+
|0xf    |0x200001 |0x0   |0x0   |0x2210 |0x1   |0x0   |0x80000e |0x27270802|
|0xb    |0x200000 |0x0   |0x0   |0x2210 |0x1   |0x0   |0x800000 |0x27270801|
+----------+----------+--------+--------+--------+-------+--------+----------+----------+
| TOTAL: 2                                                                    |
+----------+----------+--------+--------+--------+-------+--------+----------+----------+
```

9. Verify that there are no GTP parsing errors on the anchor Packet Forwarding Engine.

NOTE: Also verify that the *In* stats on the Gn (S5) interface are incrementing.

```
st@user> show vjnh 0 exceptions terse
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

Trace Data Packets from Gi to Gn Interfaces

Requirements

This example uses the following hardware and software components:

- An operational MX chassis
- Junos OS Mobility package

Tracing Data Packets

This topic shows how to trace Gi to Gn data packets.

Configuration

Setting up Data Packet Tracing

Step-by-Step Procedure

The following examples explain how to trace Gi to Gn data packets.

1. Verify that an aggregate route is installed. In this case, aggregate route 39.39.4/22 for user equipment is present in the Gi VRF (inet.0 in the following example).

```
user@host> show mobile-gateways subscribers

          MSISDN       Subscriber      Peer           APN
                       Address         Address
      1234567890        39.39.4.1   172.23.9.196   internet123
      1234567891        39.39.4.2   172.23.9.196   internet123
```

2. To see details for subscribers, enter:

```
user@host> show mobile-gateways subscribers extensive

  MSISDN : 1234567890        Subscriber Address - V4 : 39.39.4.1
    IMSI  : 22321321312336f    Control Plane Peer Address : 172.23.9.196
    NSAPI/EBI : 5              Data Plane Peer Address : 172.23.9.196
    Control TEID - Local : 8000000       Remote : 101
    Data TEID   - Local : 100000         Remote : 102
    APN name : internet123               Charging ID : 8000000
    Control - FPC : 1  PIC : 0   Anchor PFE : 129
    QCI/ARP : 5 /0     GBR: 0       MBR: 0
    Subscriber state : Established       Bearer State :   Established
                       Bearer Substate : -
    Last statistics collection time : None collected

    MSISDN : 1234567892        Subscriber Address - V4 : 39.39.4.2
    IMSI  : 22321321312337f    Control Plane Peer Address : 172.23.9.196
    NSAPI/EBI : 5              Data Plane Peer Address : 172.23.9.196
```

```
Control TEID - Local : 8000001       Remote : 103
Data TEID    - Local : 100001        Remote : 104
APN name : internet123               Charging ID : 8000001
Control - FPC : 1  PIC : 0   Anchor PFE : 129
QCI/ARP : 5 /0     GBR: 0       MBR: 0
Subscriber state : Established       Bearer State :    Established
                          Bearer Substate : -
Last statistics collection time : None collected
```

3. Since the user equipment IP address starts with **39.39**, look for aggregation routes with that prefix.

```
user@host> show route ip table index 0r

IPv4 Route Table 0, default.0, 0x0:
Destination   NH IP Addr     Type    NH ID Interface
------------  ---------------- -------- ----- ---------
default                          Reject   42
0.0.0.0                          Discard   40
10.255.15.135             10.255.15.135    Local   576
12.9.1.1                12.9.1.1         Local   645 ms-1/0/0.0
29.29.29/24                        Discard   626 mif.0
29.29.29.100              29.29.29.100     Local   625
39.39.4/22                       Unicast   677 mif.0  <---
```

4. Verify that the NH for the aggregate route uses mif ifl for the APN to which the subscriber belongs and that the NH's L2 interface corresponds to the anchor Packet Forwarding Engine.

```
user@host> show nhdb id 677 extensive

  ID    Type    Interface   Next Hop Addr   Protocol     Encap    MTU     Flags  PFE
internal Flags
----- -------- ------------- --------------- ---------- ------------ ---- ----------
--------------------
 677  Unicast  mif.0       default          IPv4   Unspecified    0  0x08000000
0x00000000

Flags:          0x08000000
PFE internal flags: 0x00000000
 L2-Interface: pfe-0/0/0.16383 (82) <--- ANCHOR PFE
```

5. Verify that the anchor Packet Forwarding Engine has the subscriber entry (in this example: LU id 0).

```
user@host> show vbf hw 0 subscriber-table downlink

+----------+--------+----------+--------+-------+--------+----------+----------+--------+
| RINDEX   | VRF ID | IPADDR   | CONFIG | FLAGS | CHRGID | CHRGADDR | TEID     |
NHID   |
+----------+--------+----------+--------+-------+--------+----------+----------+--------+
|0x4     |0x0    |0x27270401|0x2210 |0x0    |0x0    |0x800000 |0x65     |0x2ac  |
|0x3     |0x0    |0x27270402|0x2210 |0x0    |0x0    |0x80000e |0x67     |0x2ac  |
+----------+--------+----------+--------+-------+--------+----------+----------+--------+
| TOTAL: 2                                                                          |
+----------+--------+----------+--------+-------+--------+----------+----------+--------+
```

For a specific subscriber, you can verify that it uses the right peer NH.

```
user@host> show vbf hw 0 subscriber downlink id 39.39.4.1

  Key:
      Major Version: 0
      Minor Version: 2
        Overflow: 0
       TFT Rule Id: 0
        Unique Id: 0
        IPV4 ADDR: 0x27270401
  Ext Data:
       IDLE-TO PROFILE-ID: 0
         AAA PROFILE-ID: 0
               QCI: 5
            Policing: Disabled
          Reporting Stats: Disabled
          Charging Stats: Enabled
               Drop: 0
            Send to Upic: 0
               Valid: 1
               Proto: V4
            Seq Num Proc: Disabled
              Gtp Ver: 1
        Num VBF ext words: 0
          Num VBF words: 1
        Charg Stat Addr: 0x800000
       Charg Profile Id: 0
        Trigger Pending: 0
         Vol Limit Hit: 0
        Time Limit Hit: 0
       Tariff Change Hit: 0
         Delete Event: 0
         Signal Event: 0
          Tariff Id: 0
         Update First: 0
       Time Limit Check: 0
         Chrg Inst Id: 0
         Policer Type: 0
        Policer Color: 0
         Policer Oper: 0
        Policer Count: 0
      Policer Addr Offset: 0x0000
     Reporting Stats Addr: 0x000000
         Remote Index: 0x0006
          VBF Info[0]: 0x58
          VBF Info[1]: 0x17
          VBF Info[2]: 0x81
          VBF Info[3]: 0xaa
          VBF Info[4]: 0x0
          VBF Info[5]: 0x0
          VBF Info[6]: 0x0
          VBF Info[7]: 0x0
          VBF Info[8]: 0x0
          VBF Info[9]: 0x0
         VBF Info[10]: 0x0
         VBF Info[11]: 0x0
         VBF Info[12]: 0x0
```

```
        VBF Info[13]: 0x0
        VBF Info[14]: 0x0
        VBF Info[15]: 0x0
        VBF Info[16]: 0x0
        VBF Info[17]: 0x0
        VBF Info[18]: 0x0
        VBF Info[19]: 0x0
        VBF Info[20]: 0x0
        VBF Info[21]: 0x0
        VBF Info[22]: 0x0
        VBF Info[23]: 0x0
        VBF Info[24]: 0x0
        VBF Info[25]: 0x0
        VBF Info[26]: 0x0
        VBF Info[27]: 0x0
    Sideband:
    template nh vaddr: 0xd0239f
        peer nh id: 0x02ac
      udp src port: 0x00000000000000
          TEID: 0x00000000000065
      exp seq num: 0x00000000000000
```

NOTE: Peer nh id 0x02ac (684 decimal) is the NH that performs the GTP encapsulation. Template NH is the MIF OIF start. Having Zero peer nh id causes the downlink traffic to fail.

6. Verify the S-GW/SGSN IP address and other fields used in the GTP encapsulation from the peer NH.

```
user@#host> show nhdb id 684 extensive1

 ID    Type    Interface   Next Hop Addr   Protocol    Encap    MTU    Flags  PFE
internal Flags
----- -------- ------------ --------------- ---------- ------------ ---- ----------
--------------------
 684  Unicast  mif.16000    default         IPv4  Unspecified   0  0x04000000
0x00000000

Flags:          0x04000000
PFE internal flags: 0x00000000

Dram Bytes    : 268
PreComputed MTU: 0
Flags         : 0x4000000
Parent NHID   :0
      Feature List: NH
        [pfe-0]: 0x08fe5b2000010000;
        [pfe-1]: 0x08fe5b1000010000;
       f_mask:0x00600000; c_mask:0xc0000000; f_num:11; c_num:2,  inst:-1
      Idx#9     ucast:
        [pfe-0]: 0x1007f2fe00ffffff
        [pfe-1]: 0x1007f2f3c0ffffff
```

```
    Idx#10 ifl-output:
      [pfe-0]: 0x27fffff80001040c
      [pfe-1]: 0x27fffff80001040c



PFE:0

Encap-ptr chain:
----------------
Encapsulation Pointer (0x46a86d58) data:
  Encap-ptr-type:gtp
  Ucode EType:tunnel-encaps
  Ref Count:1
  Control-Word: GTP(0x03)
  Jnh-mem: size: 2; addr: 0x82
  Key: 0x0d000003ffffffff-ac1709c4ac170964401100:

 GTP Details:
     Allow Frag,

    SrcIP: 172.23.9.100  <-- GGSN/MBG1 IP address

    DstIP: 172.23.9.196 <-- SGSN/SGW IP address

     ttl: 64
     l4_proto: 17
     l3_proto: V4
 JNH words:0x2802200000030000   JNH words:0xac170964ac1709c4



PFE:1

Encap-ptr chain:
----------------
Encapsulation Pointer (0x46a86d58) data:
  Encap-ptr-type:gtp
  Ucode EType:tunnel-encaps
  Ref Count:1
  Control-Word: GTP(0x03)
  Jnh-mem: size: 2; addr: 0x84
  Key: 0x0d000003ffffffff-ac1709c4ac170964401100:

 GTP Details:
     Allow Frag,

    SrcIP: 172.23.9.100

    DstIP: 172.23.9.196

     ttl: 64
     l4_proto: 17
     l3_proto: V4
 JNH words:0x2802200000030000   JNH words:0xac170964ac1709c4
```

7. Verify that the MIF OIF features are correct.

```
user@#host> show jnh 0 vread 0xd0239f

  Addr:0xd0239f, Data = 0x08fe580000030000

  NPC0(curve vty)# sh jnh 0 decode 0x08fe580000030000

  CallNH:desc_ptr:0x1fcb00, mode=0, rst_stk=0x0, count=0x3
   0x1fcafc  0 : 0x27fffff80001640c  <-- Per subscriber Fixed classifier applied on
  MIF IFL
   0x1fcafd  1 : 0x02000fe52e000804  <-- Proto-type demux
   0x1fcafe  2 : 0x08fe50e000010000  <-- Mobile edge features (per subscriber
  policer, charging)
   0x1fcaff  3 : 0x1274040ea00003c0  <-- WAN out
```

8. Verify subscriber-fixed classifier applied on MIF IFL.

```
user@#host> show jnh 0 decode 0x27fffff80001640cf
  UcodeNH:Vbf Indirect, var-id = VBF_VAR_IFL_FIXED_CLASSIFIER(11)
```

9. Verify the demux prototype.

```
user@#host> show jnh 0 decode 0x02000fe52e000804
  IndexNH:key_ptr:0x80/0, desc_ptr=0x1fca5c, max=8, nbits=4
```

10. Verify mobile edge features (per subscriber policer, charging).

```
user@#host> show jnh 0 decode 0x08fe50e000010000

  CallNH:desc_ptr:0x1fca1c, mode=0, rst_stk=0x0, count=0x1
   0x1fca1a  0 : 0xc800000725200041
   0x1fca1b  1 : 0xc800000000000040

  NPC0(curve vty)# sh jnh 0 decode 0xc800000725200041

  JNH_ME_NH:
      opcode = 0x00000019
      desc_ptr = 0x00000000
      data = 0x39290002
      func_code = 0x00000001
  JNH_ME_NHDATA_ME_POLICER:
      normal = 0x0000e4a4
      ext_data = 0x00000000
      default = 0x00000000
      parameterized = 0x00000001
      next_nh = 0x00000000


  NPC0(curve vty)# sh jnh 0 decode 0xc800000000000040

  JNH_ME_NH:
      opcode = 0x00000019
      desc_ptr = 0x00000000
      data = 0x00000002
      func_code = 0x00000000
  JNH_ME_NHDATA_ME_CHARGING:
      report_stat_en = 0x00000000
```

```
default = 0x00000000
parameterized = 0x00000001
next_nh = 0x00000000
```

11. Verify WAN.

```
user@#host> show jnh 0 decode 0x1274040ea00003c0

ModifyNH: Subcode=SetQueue(9),Desc=0xd0103a,Data=0x3c0,NextNH=1


Dram Bytes: 440
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5
- How to Trace Data Packets from Gn to Gi Interfaces on page 23

How to Verify Charging Statistics Processing

Requirements

This example uses the following hardware and software components:

- An operational MX chassis
- Junos OS Mobility package

Verifying Packet Forwarding Engine Charging Statistics Are Processing Properly

This section shows an example of verifying that Packet Forwarding Engine charging statistics are processed from the LU to the Stats agent.

Configuration

Processing Packet Forwarding Engine Charging Statistics

Step-by-Step Procedure

The following procedure shows how to verify that charging statistics are transferred from the LU to Stats agent via TOE. The three components required to ship charging statistics from the forwarding plane to the control plane are:

- Callout thread: This is a 0.5 sec periodic thread that runs in the LU and is responsible for preparing the charging statistics.
- Charging thread: This thread runs on the LU TOE and is responsible for shipping charging statistics from the LU to the Stats agent.
- Stats agent: This runs on the Packet Forwarding Engine host CPU and is responsible for forwarding charging statistics to the charging module.

The sequence of charging statistics transfer is:

- Callout thread populates charging statistics data in the Callout FIFO.
- Callout thread triggers Charging thread to notify it of availability of data in the Callout FIFO.
- Charging thread checks whether space is available on the Stats FIFO to successfully transfer statistics from the Callout FIFO.
- If there is not enough space available on the Stats FIFO, the Charging thread aborts the statistics read from the Callout FIFO.
- If enough space is available in the Stats FIFO, then the Charging thread completes the transfer in two steps:
  - Copy statistics data from Callout FIFO to TOE Lmem.
  - DMA statistics data from TOE Lmem to Stats FIFO

Follow these steps to verify proper charging statistics handling.

1. Verify that the Callout thread is populating charging statistics in the Callout FIFO. Check the Callout FIFO descriptor to see whether the Tail (write) pointer moves. The Callout thread increments this Tail (write) pointer by the number of words that it wrote to the Callout FIFO. Therefore, if the Tail (wr) pointer moves, it means that the Callout thread is writing to the Callout FIFO.

   ```
   user@host> show jnh 0 ucode-vars

   ...
   ME CHRG information:
   Base address                    : 0xc0000026
   ME CHRG fifo tail(wr)/head(rd)  : 2/0    <================= Check if tail(wr)
   pointer moves
   ME CHRG fifo base/size          : 0x01300000/1048576
   ME CHRG next walk cookie        : 16140901064495857675
   ME CHRG time stamp              : 335007768900
   ```

NOTE: In the preceding snippet, 2/0 means that Tail (write) is 2 and Head (read) is 0. A value of 2 for Tail (write) means that the Callout thread has written two words to Callout FIFO.

2. Verify that the charging thread is being triggered by the callout thread to indicate data availability for transfer. The callout thread triggers the charging thread to notify it of availability of data in the callout FIFO. To verify that the charging thread is seeing these triggers, you could dump the TOE mobile-edge counters where the count of triggers from the callout thread is maintained. The count represents the count of triggers that the charging thread is able to honor—that is, the charging thread has determined that there are enough resources available to initiate a transfer.

```
user@host> show toe pfe 3 lu 0 mobile-edge counters

                Counter block location in LMEM: 0x4270

Host FIFO full          : 0
Callout FIFO empty      : 0
Host addr buff size unaligned : 0
Host addr page size unaligned : 53
Test Reg 1              : 0
Test Reg 2              : 0
Callout-to-LMEM copy bytes   : 231368
LMEM-to-Host dma bytes       : 231368
Callout triggers        : 2675      <=== count of triggers from Callout thread
```

3. Verify that the Stats FIFO is full. The charging thread checks whether space is available on the Stats FIFO to successfully transfer statistics from the callout FIFO. If the Stats FIFO is full, the transfer is not initiated. For each trigger from the callout thread, the charging thread checks the status of the Stats FIFO and if it is full, increments the counter.

```
user@host> show toe pfe 3 lu 0 mobile-edge counters

Counter block location in LMEM: 0x4270

Host FIFO full          : 0   <=== increments on each trigger from
                                Callout thread, if Stats FIFO is FULL and
                                 trigger cannot be "honored"
Callout FIFO empty      : 0
Host addr buff size unaligned : 0
Host addr page size unaligned : 53
Test Reg 1              : 0
Test Reg 2              : 0
Callout-to-LMEM copy bytes   : 231368
LMEM-to-Host dma bytes       : 231368
Callout triggers        : 2675
```

4. Verify that the charging thread is reading the charging statistics from the callout FIFO. Check the TOE mobile-edge counters to get the total number of “bytes” of

charging data transferred by the Charging thread, from the Callout FIFO to the TOE LMem.

```
user@host> show toe pfe 3 lu 0 mobile-edge counters

  Counter block location in LMEM: 0x4270

  Host FIFO full          : 0
  Callout FIFO empty      : 0
  Host addr buff size unaligned : 0
  Host addr page size unaligned : 53
  Test Reg 1              : 0
  Test Reg 2              : 0
  Callout-to-LMEM copy bytes   : 231368   <== Number of bytes transferred from
                                             Callout FIFO to TOE LMem
  LMEM-to-Host dma bytes      : 231368
  Callout triggers        : 2675
```

5. Verify that the charging thread is sending charging data to the Stats FIFO. Check the TOE mobile-edge counters to get the number of “bytes” of charging statistics transferred by the charging thread, from the TOE LMem to Stats FIFO.

```
user@host> show toe pfe 3 lu 0 mobile-edge counters

  Counter block location in LMEM: 0x4270

  Host FIFO full          : 0
  Callout FIFO empty      : 0
  Host addr buff size unaligned : 0
  Host addr page size unaligned : 53
  Test Reg 1              : 0
  Test Reg 2              : 0
  Callout-to-LMEM copy bytes   : 231368
  LMEM-to-Host dma bytes      : 231368      <=== Number of bytes transferred by
                                             charging thread from TOE LMem to Stats
  FIFO
  Callout triggers        : 2675
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Monitoring Resources on page 5

Example: Monitoring a P-GW with Call Trace

This example shows how to monitor MobileNext Broadband Gateway Packet Data Network Gateway (P-GW) operation with call trace.

- Requirements on page 38
- Overview on page 38
- Configuration on page 38

Requirements

This example uses the following hardware and software components:

- An installed and operational MobileNext Broadband Gateway chassis
- The properly installed and Operational Junos OS MobileNext Broadband Gateway software packages

Before you use call trace for monitoring, be sure you have:

- Made sure that this P-GW is configured correctly

Overview

This example shows how to monitor the P-GW operation using call trace.

Topology

This procedure is independent of other network devices.

Configuration

To use call trace monitoring, perform these tasks:

- Monitoring with Call Trace on page 38

Monitoring with Call Trace

Step-by-Step Procedure

To use call trace monitoring:

1. Start the call trace.

```
user@MGB1> request unified-edge ggsn-pgw call-trace start imsi 101313783444554
```

NOTE: This example uses IMSI number as the basis for the call trace.

```
Service PIC    Status
ms-0/0/0         success
ms-10/0/0     success
```

2. Display the call trace information.

```
user@MGB1> request unified-edge ggsn-pgw call-trace show detail

  Call trace information :

   Identifier : call_trace_id_1       Trace file : call_trace_id_1_01142012_152946
   Status : not-done    Create Mask : 0x44      Complete Mask : 0x0
   IMSI : 505002003476097
   Calls Traced : 1
```

3. Stop the call trace. You can stop all traces at once, as shown here.

```
user@MGB1> request unified-edge ggsn-pgw call-trace stop all

  Service PIC    Status
  ms-0/0/0     success
  ms-10/0/0     success
```

4. Display the available trace files.

```
user@MGB1> request unified-edge ggsn-pgw call-trace show

  Identifier        File name              Status      SPIC Mask   SPIC Mask
                                          create      complete
  call_trace_id_1 call_trace_id_1_01142012_152946      done    0x44    0x44
  {master}
```

5. Display the contents of a call trace log file. Just as Syslog files, call traces are stored in **/var/log**.

```
user@MGB1> request unified-edge ggsn-pgw call-trace show
  /var/log/call_trace_id_1_01142012_152946

  Dec  5 16:12:30.1118022 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Encoding GTP
   V1 Header
  Dec  5 16:12:30.1118044 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Msg Type 17
  SeqNumber 584 TEID 302149215
  Dec  5 16:12:30.1118067 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Encoding
  CAUSE IE
  Dec  5 16:12:30.1118088 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Encoding cause
   Authentication failure
  Dec  5 16:12:30.1118109 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: ERROR: Non
  successful cause value(208) sent
  Dec  5 16:12:30.1118131 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: ENCODED Error
   Response of length 14 for msg 17


  Nov 26 16:20:53.1327494 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 change reporting
  action not supported
  Nov 26 16:20:53.1327997 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 Handler returned
  PASS
  Nov 26 16:20:53.1328020 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 calling afProcessEvent
   for event=sessUpdate, state=Established
  Nov 26 16:20:53.1328042 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 i = 0, size = 1,
  evtProtocolBitmap = 19 info->cause = 0, info->sub_cause = 0
  Nov 26 16:20:53.1328068 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 calling session-evt
  handler #0 for state=Established, event=sessUpdate info->cause = 0,
  info->subcause=0, num_handlers=1
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Call Trace Overview on page 4
- Example: Monitoring an S-GW with Call Trace on page 40

Example: Monitoring an S-GW with Call Trace

This example shows how to monitor MobileNext Broadband Gateway Packet Data Network Gateway (S-GW) operation with call trace.

Requirements

This example uses the following hardware and software components:

- An installed and operational MobileNext Broadband Gateway chassis
- The properly installed and Operational Junos OS MobileNext Broadband Gateway software packages

Before you use call trace for monitoring, be sure you have:

- Made sure that this S-GW is configured correctly

Overview

This example shows how to monitor the P-GW operation using call trace.

Topology

This procedure is independent of other network devices.

Configuration

To use call trace monitoring, perform these tasks:

Monitoring with Call Trace

Step-by-Step Procedure

To use call trace monitoring:

1. Start the call trace.

```
user@MGB1> request unified-edge sgw call-trace start fpc-slot 4 pic-slot 0 next-call
  10
```

NOTE: This example uses FPC and PIC and next-call option as the basis for the call trace.

```
Service PIC      Status
```

```
ms-0/0/0  success
ms-1/0/0  success
```

2. Display the call trace information.

```
user@MGB1> rrequest unified-edge sgw call-trace show brief

Call trace information :
 Identifier : call_trace_id_10    Trace file :
call_trace_id_10_02112012_205634
 Status : done    Create Mask : 0x0   Complete Mask : 0x0
 Next Call : 10
 Calls Traced : 0      FPC : 5   PIC : 0
Identifier : call_trace_id_11    Trace file :
call_trace_id_11_02112012_205932
Status : done    Create Mask : 0x40   Complete Mask : 0x40
Calls Traced : 0
Identifier : call_trace_id_12    Trace file :
call_trace_id_12_02112012_210001
Status : not-done    Create Mask : 0x40  Complete Mask : 0x0
Next Call : 5
Calls Traced : 0      FPC : 4   PIC : 0
Identifier : call_trace_id_13    Trace file :
call_trace_id_13_02112012_210353
Status : duplicate    Create Mask : 0x0   Complete Mask : 0x0
Next Call : 5
Calls Traced : 0      FPC : 4   PIC : 0
```

3. Stop the call trace. You can stop all traces at once, as shown here.

```
user@MGB1> request unified-edge sgw call-trace stop all

 Service PIC    Status
 ms-0/0/0       success
 ms-1/0/0     success
```

4. Display the available trace files.

```
user@MGB1> request unified-edge sgw call-trace show

 Identifier        File name             Status     SPIC Mask   SPIC Mask
                                        create     complete
 call_trace_id_1 call_trace_id_1_01142012_152946     done    0x44    0x44
{master}
```

5. Display the contents of a call trace log file. Just as Syslog files, call traces are stored in **/var/log**.

```
user@MGB1> request unified-edge sgw call-trace show
  /var/log/call_trace_id_1_01142012_152946

Dec  5 16:12:30.1118022 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Encoding GTP
 V1 Header
Dec  5 16:12:30.1118044 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Msg Type 17
SeqNumber 584 TEID 302149215
Dec  5 16:12:30.1118067 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Encoding
CAUSE IE
Dec 5 16:12:30.1118088 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: Encoding cause
```

```
 Authentication failure
Dec  5 16:12:30.1118109 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: ERROR: Non
successful cause value(208) sent
Dec  5 16:12:30.1118131 cpu_id: [6] gtid: [9] tid: [2] app_id: 1 Encode: ENCODED Error
 Response of length 14 for msg 17


Nov 26 16:20:53.1327494 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 change reporting
action not supported
Nov 26 16:20:53.1327997 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 Handler returned
PASS
Nov 26 16:20:53.1328020 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 calling afProcessEvent
 for event=sessUpdate, state=Established
Nov 26 16:20:53.1328042 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 i = 0, size = 1,
evtProtocolBitmap = 19 info->cause = 0, info->sub_cause = 0
Nov 26 16:20:53.1328068 cpu_id: [5] gtid: [8] tid: [1] app_id: 1 calling session-evt
handler #0 for state=Established, event=sessUpdate info->cause = 0,
info->subcause=0, num_handlers=1
```

Related Documentation

- Monitoring the Mobile Environment - Key Performance Indicators on page 3
- Call Trace Overview on page 4
- Example: Monitoring a P-GW with Call Trace on page 37

CHAPTER 5

Operational Commands

request unified-edge ggsn-pgw call-trace clear

Syntax **request unified-edge ggsn-pgw call-trace clear**

Release Information Command introduced in Junos OS Mobility Release 11.2W.

Description Clear the completed or duplicate subscriber call traces on one or more Gateway GPRS Support Nodes (GGSNs) or Packet Data Network Gateways (P-GWs).

Options This command has no options.

Required Privilege Level unified-edge

Related Documentation

- request unified-edge ggsn-pgw call-trace show on page 45
- request unified-edge ggsn-pgw call-trace start on page 48
- request unified-edge ggsn-pgw call-trace stop on page 50

List of Sample Output request unified-edge ggsn-pgw call-trace on page 44

Output Fields No message is displayed on successful execution of this command; otherwise an error message is displayed.

Sample Output

request unified-edge ggsn-pgw call-trace

```
user@host> request unified-edge ggsn-pgw call-trace clear
```

request unified-edge ggsn-pgw call-trace show

Syntax

request unified-edge ggsn-pgw call-trace show
<all | completed | current>
<brief | detail>

Release Information

Command introduced in Junos OS Mobility Release 11.2W.

Description

Display the information related to subscriber call tracing on one or more Gateway GPRS Support Nodes (GGSNs) or Packet Data Network Gateways (P-GWs).

Options

none—(Same as brief) Display the information related to subscriber call tracing in brief.

all | completed | current—(Optional)Display the call trace information for the following:

- **all**—All calls.
- **completed**—Completed calls only.
- **current**—Call traces that are currently active.

brief | detail—(Optional) Display the specified level of output.

Required Privilege Level

unified-edge

Related Documentation

- request unified-edge ggsn-pgw call-trace clear on page 44
- request unified-edge ggsn-pgw call-trace start on page 48
- request unified-edge ggsn-pgw call-trace stop on page 50

List of Sample Output

request unified-edge ggsn-pgw call-trace show brief on page 46
request unified-edge ggsn-pgw call-trace show detail on page 46

Output Fields

Table 3 on page 45 lists the output fields for the **request unified-edge ggsn-pgw call-trace show** command. Output fields are listed in the approximate order in which they appear.

Table 3: request unified-edge ggsn-pgw call-trace show Output Fields

Field Name	Field Description	Level of Output
Identifier	Identifier for the call trace.	All levels
File name or **Trace file**	Name of the call trace file.	All levels
Status	Status of the call trace: • **done**—Call trace complete. • **not-done**—Call trace in progress. • **duplicate**—Another call trace record is present that has the same attributes.	All levels

Table 3: request unified-edge ggsn-pgw call-trace show Output Fields *(continued)*

Field Name	Field Description	Level of Output
SPIC Mask Create or **Create Mask**	Internal mask of the services PIC where this call trace was enabled.	All levels
SPIC Mask Complete or **Complete Mask**	Internal mask of the services PIC where this call trace was completed.	All levels
IMSI	International Mobile Subscriber Identity (IMSI) of the subscriber's user equipment (UE).	
MSISDN	Mobile station ISDN (MSISDN) of the subscriber's user equipment.	
Calls Traced	Number of calls traced.	**detail**
Next Call	Number of next calls to be traced. For example, a value of 10 indicates that the next 10 calls are traced.	**detail**
APN	Access Point Name (APN) pertaining to the subscriber's call.	**detail**
FPC	FPC slot on which the call trace was enabled. This field is displayed only if the call trace is enabled on the FPC slot.	**detail**
PIC	PIC slot on which the call trace was enabled. This field is displayed only if the call trace is enabled on the PIC slot.	**detail**

Sample Output

request unified-edge ggsn-pgw call-trace show brief

```
user@host> request unified-edge ggsn-pgw call-trace show brief
  Identifier           File name           Status       SPIC Mask   SPIC Mask
                                                         create      complete
 call_trace_id_2 call_trace_id_2_02112012_060450        done 0x10   0x10
 call_trace_id_3 call_trace_id_3_02112012_070614        done 0x10   0x10
 call_trace_id_4 call_trace_id_4_02112012_071342   duplicate 0x0    0x0
 call_trace_id_5 call_trace_id_5_02112012_201317   duplicate 0x0    0x0
 call_trace_id_6 call_trace_id_6_02112012_201649   duplicate 0x0    0x0
 call_trace_id_7 call_trace_id_7_02112012_202501        done 0x0    0x0
 call_trace_id_8 call_trace_id_8_02112012_204718   duplicate 0x0    0x0
 call_trace_id_9 call_trace_id_9_02112012_204759    not-done 0x10   0x0
```

request unified-edge ggsn-pgw call-trace show detail

```
user@host> request unified-edge ggsn-pgw call-trace show detail
 Call trace information :

  Identifier : call_trace_id_13        Trace file :
call_trace_id_13_02292012_001343
  Status : not-done    Create Mask : 0x200      Complete Mask : 0x0
  IMSI : 29299
  Calls Traced : 0
  Identifier : call_trace_id_14        Trace file :
call_trace_id_14_02292012_001348
  Status : not-done    Create Mask : 0x200      Complete Mask : 0x0
                                                      MS-ISDN: 2929910000000000
```

```
  Calls Traced : 0
  Identifier : call_trace_id_15        Trace file :
call_trace_id_15_02292012_001408
  Status : not-done    Create Mask : 0x200      Complete Mask : 0x0
  Next Call : 1                        APN : jnpr-sunnyvale
  Calls Traced : 0
  Identifier : call_trace_id_16        Trace file :
call_trace_id_16_02292012_001416
  Status : not-done    Create Mask : 0x200      Complete Mask : 0x0
  Calls Traced : 0                     FPC : 3   PIC : 1
  Identifier : call_trace_id_17        Trace file :
call_trace_id_17_02292012_001424
  Status : done        Create Mask : 0x200      Complete Mask : 0x200
  Next Call : 2
  Calls Traced : 2
```

request unified-edge ggsn-pgw call-trace start

Syntax

request unified-edge ggsn-pgw call-trace start
<apn-name *name*>
<fpc-slot *slot*>
<imsi *imsi*>
<msisdn *msisdn*>
<next-call *next-call*>
<pic-slot *slot*>

Release Information

Command introduced in Junos OS Mobility Release 11.2W.

Description

Start the subscriber call tracing on one or more Gateway GPRS Support Nodes (GGSNs) or Packet Data Network Gateways (P-GWs).

Options

none—Start the subscriber call tracing.

apn-name *apn-name*—(Optional) Start the call tracing for subscribers accessing the specified access point name (APN).

fpc-slot *slot*—(Optional) Start the call tracing for subscribers on the specified FPC slot.

imsi *imsi*—(Optional) Start the call tracing for subscribers with the specified International Mobile Subscriber Identity (IMSI) number.

msisdn *msisdn*—(Optional) Start the call tracing for subscribers with the specified Mobile station ISDN (MSIDSN) number.

next-call *next-call*—(Optional) Start the call tracing for the specified number of next call events (1 through 50). For example, if you specify 10, then the next 10 calls will be traced.

pic-slot *slot*—(Optional) Start the call tracing for subscribers on the specified PIC slot. You must specify an FPC slot before specifying a PIC slot number.

Required Privilege Level

unified-edge

Related Documentation

- request unified-edge ggsn-pgw call-trace clear on page 44
- request unified-edge ggsn-pgw call-trace show on page 45
- request unified-edge ggsn-pgw call-trace stop on page 50

List of Sample Output

request unified-edge ggsn-pgw call-trace start fpc-slot 5 pic-slot 0 next-call 10 on page 49

Output Fields

Table 4 on page 49 lists the output fields for the **request unified-edge ggsn-pgw call-trace start** command. Output fields are listed in the approximate order in which they appear.

Table 4: request unified-edge ggsn-pgw call-trace start Output Fields

Field Name	Field Description
Session PIC	Session PIC for which the call trace status is displayed.
Status	Status of the call trace: • **duplicate**—Another call trace record is present that has the same attributes. • **success**—Call trace started successfully. • **fail**—Call tracing could not be started.

Sample Output

request unified-edge ggsn-pgw call-trace start fpc-slot 5 pic-slot 0 next-call 10

```
user@host> request unified-edge ggsn-pgw call-trace start fpc-slot 5 pic-slot 0 next-call 10
        Session PIC              Status
        ms-0/1/0                 success
        ms-1/1/0                 success
```

request unified-edge ggsn-pgw call-trace stop

Syntax request unified-edge ggsn-pgw call-trace stop
<all>
<identifier *call-trace-identifier*>

Release Information Command introduced in Junos OS Mobility Release 11.2W.

Description Stop the previously configured subscriber call tracing on one or more Gateway GPRS Support Nodes (GGSNs) or Packet Data Network Gateways (P-GWs).

Options

none—(Same as all) Stop all the subscriber call tracing options.

all—(Optional) Stop all the subscriber call tracing operations.

identifier *identifier*—(Optional) Stop the call tracing for the specified call trace identifier.

Required Privilege Level unified-edge

Related Documentation

- request unified-edge ggsn-pgw call-trace clear on page 44
- request unified-edge ggsn-pgw call-trace show on page 45
- request unified-edge ggsn-pgw call-trace start on page 48

List of Sample Output request unified-edge ggsn-pgw call-trace stop on page 50

Output Fields Table 5 on page 50 lists the output fields for the **request unified-edge ggsn-pgw call-trace stop** command. Output fields are listed in the approximate order in which they appear.

Table 5: request unified-edge ggsn-pgw call-trace stop Output Fields

Field Name	Field Description
Session PIC	Session PIC for which the call trace status is displayed.
Status	Status of the call trace: • **success**—Call trace stopped successfully. • **fail**—Call tracing could not be stopped.

Sample Output

request unified-edge ggsn-pgw call-trace stop

```
user@host> request unified-edge ggsn-pgw call-trace stop
       Session PIC            Status
       ms-0/1/0             success
       ms-1/1/0             success
```

request unified-edge sgw call-trace clear

Syntax **request unified-edge sgw call-trace clear**

Release Information Command introduced in Junos OS Mobility Release 11.4W.

Description Clear the completed or duplicate subscriber call traces on one or more Serving Gateways (S-GWs).

Options This command has no options.

Required Privilege Level unified-edge

Related Documentation

- request unified-edge sgw call-trace show on page 52
- request unified-edge sgw call-trace start on page 55
- request unified-edge sgw call-trace stop on page 57

List of Sample Output request unified-edge sgw call-trace clear on page 51

Output Fields No message is displayed on successful execution of this command; otherwise an error message is displayed.

Sample Output

request unified-edge sgw call-trace clear

```
user@host> request unified-edge sgw call-trace clear
```

request unified-edge sgw call-trace show

Syntax
request unified-edge sgw call-trace show
<all | completed | current>
<brief | detail>

Release Information
Command introduced in Junos OS Mobility Release 11.4W.

Description
Display the information related to subscriber call tracing on one or more Serving Gateways (S-GWs).

Options
none—(Same as brief) Display the information related to subscriber call tracing in brief.

all | completed | current—(Optional)Display the call trace information for the following:

- **all**—All calls.
- **completed**—Completed calls only.
- **current**—Call traces that are currently active.

brief | detail—(Optional) Display the specified level of output.

Required Privilege Level
unified-edge

Related Documentation
- request unified-edge sgw call-trace clear on page 51
- request unified-edge sgw call-trace start on page 55
- request unified-edge sgw call-trace stop on page 57

List of Sample Output
request unified-edge sgw call-trace show brief on page 53
request unified-edge sgw call-trace show detail on page 53

Output Fields
Table 6 on page 52 lists the output fields for the **request unified-edge sgw call-trace show** command. Output fields are listed in the approximate order in which they appear.

Table 6: request unified-edge sgw call-trace show Output Fields

Field Name	Field Description	Level of Output
Identifier	Identifier for the call trace.	All levels
File name or **Trace file**	Name of the call trace file.	All levels
Status	Status of the call trace: • **done**—Call trace complete. • **not-done**—Call trace in progress. • **duplicate**—Another call trace record is present that has the same attributes.	All levels

Table 6: request unified-edge sgw call-trace show Output Fields *(continued)*

Field Name	Field Description	Level of Output
SPIC Mask Create or **Create Mask**	Internal mask of the services PIC where this call trace was enabled.	All levels
SPIC Mask Complete or **Complete Mask**	Internal mask of the services PIC where this call trace was completed.	All levels
IMSI	International Mobile Subscriber Identity (IMSI) of the subscriber's user equipment (UE).	
MSISDN	Mobile station ISDN (MSISDN) of the subscriber's user equipment.	
Calls Traced	Number of calls traced.	**detail**
Next Call	Number of next calls to be traced. For example, a value of 10 indicates that the next 10 calls are traced.	**detail**
FPC	FPC slot on which the call trace was enabled. This field is displayed only if the call trace is enabled on the FPC slot.	**detail**
PIC	PIC slot on which the call trace was enabled. This field is displayed only if the call trace is enabled on the PIC slot.	**detail**

Sample Output

request unified-edge sgw call-trace show brief

```
user@host> request unified-edge sgw call-trace show brief
  Identifier           File name           Status       SPIC Mask   SPIC Mask
                                                         create      complete
 call_trace_id_10 call_trace_id_10_02112012_205634       done 0x0  0x0
 call_trace_id_11 call_trace_id_11_02112012_205932       done 0x40 0x40
 call_trace_id_12 call_trace_id_12_02112012_210001   not-done 0x40 0x0
 call_trace_id_13 call_trace_id_13_02112012_210353  duplicate 0x0  0x0
```

request unified-edge sgw call-trace show detail

```
user@host> request unified-edge sgw call-trace show detail
 Call trace information :

  Identifier : call_trace_id_10        Trace file :
call_trace_id_10_02112012_205634
  Status : done        Create Mask : 0x0        Complete Mask : 0x0
  Next Call : 10
  Calls Traced : 0                     FPC : 5   PIC : 0
  Identifier : call_trace_id_11        Trace file :
call_trace_id_11_02112012_205932
  Status : done        Create Mask : 0x40       Complete Mask : 0x40
  Calls Traced : 0
  Identifier : call_trace_id_12        Trace file :
call_trace_id_12_02112012_210001
  Status : not-done    Create Mask : 0x40       Complete Mask : 0x0
  Next Call : 5
  Calls Traced : 0                     FPC : 4   PIC : 0
  Identifier : call_trace_id_13        Trace file :
call_trace_id_13_02112012_210353
```

```
Status : duplicate   Create Mask : 0x0        Complete Mask : 0x0
Next Call : 5
Calls Traced : 0                    FPC : 4   PIC : 0
```

request unified-edge sgw call-trace start

Syntax

request unified-edge sgw call-trace start
<fpc-slot *slot*>
<imsi *imsi*>
<msisdn *msisdn*>
<next-call *next-call*>
<pic-slot *slot*>

Release Information

Command introduced in Junos OS Mobility Release 11.4W.

Description

Start the subscriber call tracing on one or more Serving Gateways (S-GWs).

Options

none—Start the subscriber call tracing.

fpc-slot *slot*—(Optional) Start the call tracing for subscribers on the specified FPC slot.

imsi *imsi*—(Optional) Start the call tracing for subscribers with the specified International Mobile Subscriber Identity (IMSI) number.

msisdn *msisdn*—(Optional) Start the call tracing for subscribers with the specified Mobile station ISDN (MSIDSN) number.

next-call *next-call*—(Optional) Start the call tracing for the specified number of next call events (1 through 50). For example, if you specify 10, then the next 10 calls will be traced.

pic-slot *slot*—(Optional) Start the call tracing for subscribers on the specified PIC slot. You must specify an FPC slot before specifying a PIC slot number.

Required Privilege Level

unified-edge

Related Documentation

- request unified-edge sgw call-trace clear on page 51
- request unified-edge sgw call-trace show on page 52
- request unified-edge sgw call-trace stop on page 57

List of Sample Output

request unified-edge sgw call-trace start fpc-slot 4 pic-slot 0 next-call 10 on page 56

Output Fields

Table 7 on page 55 lists the output fields for the **request unified-edge sgw call-trace start** command. Output fields are listed in the approximate order in which they appear.

Table 7: request unified-edge sgw call-trace start Output Fields

Field Name	Field Description
Session PIC	Session PIC for which the call trace status is displayed.

Table 7: request unified-edge sgw call-trace start Output Fields *(continued)*

Field Name	Field Description
Status	Status of the call trace: • **duplicate**—Another call trace record is present that has the same attributes. • **success**—Call trace started successfully. • **fail**—Call tracing could not be started.

Sample Output

request unified-edge sgw call-trace start fpc-slot 4 pic-slot 0 next-call 10

```
user@host> request unified-edge sgw call-trace start fpc-slot 4 pic-slot 0 next-call 10
       Session PIC            Status
       ms-0/0/0             success
       ms-1/0/0             success
```

request unified-edge sgw call-trace stop

Syntax
request unified-edge sgw call-trace stop
<all>
<identifier *call-trace-identifier*>

Release Information
Command introduced in Junos OS Mobility Release 11.4W.

Description
Stop the previously configured subscriber call tracing on one or more Serving Gateways (S-GWs).

Options
none—(Same as all) Stop all the subscriber call tracing options.

all—(Optional) Stop all the subscriber call tracing operations.

identifier *identifier*—(Optional) Stop the call tracing for the specified call trace identifier.

Required Privilege Level
unified-edge

Related Documentation
- request unified-edge sgw call-trace clear on page 51
- request unified-edge sgw call-trace show on page 52
- request unified-edge sgw call-trace start on page 55

List of Sample Output
request unified-edge sgw call-trace stop on page 57

Output Fields
Table 8 on page 57 lists the output fields for the **request unified-edge sgw call-trace stop** command. Output fields are listed in the approximate order in which they appear.

Table 8: request unified-edge sgw call-trace stop Output Fields

Field Name	Field Description
Session PIC	Session PIC for which the call trace status is displayed.
Status	Status of the call trace: • **success**—Call trace stopped successfully. • **fail**—Call tracing could not be stopped.

Sample Output

request unified-edge sgw call-trace stop

```
user@host> request unified-edge sgw call-trace stop
       Session PIC            Status
       ms-0/0/0             success
       ms-1/0/0             success
```

PART 3

Troubleshooting

CHAPTER 6

Troubleshooting Procedures

Troubleshooting Overload Conditions in the Mobile Network

The common causes of an overload condition are:

- An external server (RADIUS, DHCP, charging gateway, PCRF, and so on) is down for an extended period of time
- A burst of GTP control messages from a rebooted peer
- Capacity overload due to oversubscribed system limits
- Management operations such as bulk session deletes
- Peer reboot leading to bulk deletes resulting in higher CPU consumption

Related Documentation

- Troubleshooting Mobility on page 15

Troubleshooting Multilevel Overload Protection

To troubleshoot multilevel overloads, consider:

- Configurable low and high thresholds in percentage for each resource monitored
- Configurable Local policy to apply when the resource low or high threshold is reached

NOTE: For example: When memory usage reaches 70%, accept only calls with an allocation and retention priority (ARP) of 5 and higher. When memory usage reaches 90%, accept only calls with ARP of 3 or higher.

- Internal redirection policy to equally distribute calls to various session PICs in the chassis

Related Documentation

- Troubleshooting Mobility on page 15
- Responding to an Overload on page 62

Responding to an Overload

To respond to an overload condition, consider:

- Apply gating to incoming calls and service high-priority subscribers
- Generate alarms, traps, and logs to notify the operator
- Throttle request generated toward external entities
- Configurable redirection policy to forward calls matching a certain criteria to an external gateway
- Configurable priority-level (ARP) to service during overload condition
- Each component dynamically reports load to the central resource controller for real-time admission control.

These are default actions that the gateway performs when overload conditions occur.

Related Documentation

- Troubleshooting Mobility on page 15

Troubleshooting GTP

This topic explains how to troubleshoot GTP connectivity problems.

Since the mobility environment is a distributed system, troubleshooting this environment means you need to know details about subscribers, the SPIC/PFE which hosts the subscriber, and which SPIC is the designated SPIC for the GTP peer.

The show commands listed in here are the primary troubleshooting tools you can use to identify problems in your mobility environment.

In this scenario, the SGW is sending GTP packets, but the PGW is silently dropping them somewhere. The packets are reaching the SPIC, but they are not reaching GTP.

1. Check if packets are being dropped in the PFE. You can determine that by seeing if there are packet statistics on the PFE. If not, then packets are not getting this far. You also need to examine the packet steering filters on the PFE.
2. Check if packets have reached the SPIC. To do this, look at the packet counters on the ms0 interface.
3. Check if packets have reached the GTP stack. To do this, use:

   user@host> **show unifed-edge ggsn-pgw gtp statistics detail**

   If the packet has been received on the GTP stack, it should show up in the Rx counters.

The GTP stack always counts GTP requests and sends a response. So if a packet is received, it will be counted.

4. Verify if GTP is generating response packets and they are being acknowledged. To do this, use:

   user@host> **show unifed-edge ggsn-pgw gtp statistics detail**

   Check if the Tx counters are increasing. If they are, check the ms0 packet counters and verify that they are also increasing.

5. If the ms0 packet statistics are good, then examine the packet steering filter counters. You should see response packet counters incrementing.

In this scenario, the SGW is sending GTP packets but the PGW is returning errors.

You should check the GTP traces, because the GTP stack:

- Handles all the receiving, decoding, and encoding of GTP packets
- Performs syntax checks on IEs
- Checks for objects like APN configurations
- Manages conflicts between GTP peer versions and GTP packets

In this scenario, sessions are set up successfully, but they are being deleted over time. The likely cause is GTP path management clearing sessions. GTP path management is likely:

- Discovering that the peer is dead because of an echo request timeout.
- Restart count in recovery. IE is different from earlier messages.

1. To examine the problem, enter:

   user@host> **show unifed-edge ggsn-pgw gtp statistics detail**

   From the data returned, you should be able to see the number of echo requests (Tx) and echo responses (Rx). If Rx is smaller than Tx, then you can conclude that echo responses are not getting to the PGW. To troubleshoot this problem, do the following:

   - Check the statistics on the GTP peer to see if it is receiving echo requests. If not, debug why packets are not being sent out or are not being received by the GTP peer.
   - If the GTP peer is receiving but not responding, check the statistics/traces on the GTP peer.
   - If the GTP peer is sending back an echo response, check if it is being sent from the PFE to the SPIC.
   - If echo responses are going to the SPIC, check the packet steering filters.
   - Check the GTP traces to see if the GTP stack is receiving echo responses but not decoding or accepting them.

2. Enable GTP history so peer statistics are not deleted after sessions are deleted. To do this, user:

```
user@host> show unifed-edge ggsn-pgw gtp peer history
```

3. Check how often the peer has been declared down or dead and for what reason. If the peer was down because of a mismatching restart court, this will appear in the stats.
4. Find out why there was a mismatch in the restart count.
5. You can workaround a mismatch in the restart count by disabling path management on the PGW. This isolates the real issue behind this behavior. Once the issue is resolved, you can enable path management.

In this scenario, you experience a random set up failure.

1. To dump traces for the next n calls to help isolate this problem, use this command:

```
user@host> monitor unifed-edge ggsn-pgw call-trace start next-call
```

In this scenario, you have subscriber failures.

1. To troubleshoot this problem, turn on tracing for the subscriber who is having problems. To do this, enter:

```
user@host> monitor unifed-edge ggsn-pgw call-trace start IMSI
```

   This will output traces for this IMSI/MSISDN to help isolate the problem.

In this scenario, the system is behaving erratically when subscriber counts are high. To troubleshoot this type of problem, you need to try to isolate the problem to an APN, a SPIC, the IMSI, MSISDN, or the GTP SGW peer.

1. When you have isolated the problem, look at the GTP global statistics to determine which counter is increasing. To do this, use:

```
user@host> show unifed-edge ggsn-pgw statistics ?
```

   This presents a wealth of options for looking at the internals of the GTP stack.

   Also look at the App-fw stats/traces to see what is causing session setup failures.

In general, troubleshooting GTP makes use of the following show and trace commands:

- show subscribers gateway

```
user@host> show unifed-edge ggsn-pgw subscribers gateway gateway-name ?
```

- show gtp peer

```
user@host> show unifed-edge ggsn-pgw gtp peer ?
```

- show gtp statistics

```
user@host> show unifed-edge ggsn-pgw gtp statistics ?
```

- trace gtp

```
user@host> edit unifed-edge gateway ggsn-pgw pgw-name gtp traceoptions

{
file gtp_log size 100m;
level all
```

```
flag all
}
```

The show output will reveal the counters being incriminated which helps with isolating problems. The trace output shows the GTP encoding/decoding details.

Related Documentation

- Troubleshooting Mobility on page 15

Troubleshooting Alarms, Logs, and Traps

The mobility system generates and retains the following congestion statistics:

Current congestion status peak congestion hits

- Time when the last congestion occurred
- Duration the last congestion lasted
- Number of calls rejected during congestion
- SNMP traps
- System logs

The following are GTP traps:

- jnxMbgPgwGtpPeerGWUpNotif

  The state of a GTP peer (control or data) has changed to UP. The GTP peer in trap is specified as “Rem: 200.6.1.2 Loc: 200.6.88.1 vrf: 0 [Ctrl]" where ‘Rem’ is the remote IP address, ‘Loc’ is the local IP address, and ‘vrf’ is the vrf instance. ‘Ctrl’ indicates that this is a GTP-C peer. ‘For the GTP-U peer, string ‘Data’ is present in place of ‘Ctrl’. This trap is generated only if GTP path management for the GTP peer is enabled.

- jnxMbgPgwGtpPeerDownNotif

  The state of a GTP peer (control or data) has changed to DOWN. The GTP peer in trap is specified as “Rem: 200.6.1.2 Loc: 200.6.88.1 vrf: 0 [Ctrl]" where ‘Rem’ is the remote IP address, ‘Loc’ is the local IP address,and ‘vrf’ is the vrf instance. ‘Ctrl’ indicates that this is GTP-C peer. For GTP-U peer, string ‘Data’ will be present in place of ‘Ctrl’. This trap is generated only if GTP path management for the GTP peer is enabled.

- jnxMbgPgwGtpPeerDNThresPerPeerNotif

  The total number of GTP peer (control or data) down events per GTP peer have crossed a threshold. The GTP peer in trap is specified as “Rem: 200.6.1.2 Loc: 200.6.88.1 vrf: 0 [Ctrl]" where ‘Rem’ is remote IP address, ‘Loc’ is local IP address, ‘vrf’ is vrf instance. ‘Ctrl’ indicates that this is GTP-C peer. For GTP-U peer, string ‘Data’ is present in place of ‘Ctrl’. If the number becomes higher than the “raise threshold,” then value 1 in field jnxMbgPgwGtpAlarmState indicates the alarm-state “raised,” and if the number becomes less than “clear threshold,”’ then value 0 in field jnxMbgPgwGtpAlarmState indicates alarm-state “cleared”. This trap is generated only if GTP path management for the GTP peer is enabled.

- jnxMbgPgwGtpNumDiscardedGtpcPktThresNotif

- The following are the Subscriber Manager traps:
  - jnxMbgPgwSMGtpEventNotif

    An important GTP event has occurred. jnxMbgPgwSMGTPEventType indicates the type of event (for example, “PDP_CTXT_CREATE_REJECT”) and jnxMbgPgwSMGTPEventCause indicates the cause of the event (for example, “RESOURCE_ERR”).
  - jnxMbgPgwSMSubscribersThresGblNotif The total number of subscribers in the system has crossed a threshold.

    

    NOTE: For this trap and all remaining Subscriber Manager traps, two thresholds (“High” and “Low”) have been defined. For each threshold, this notification is generated when a threshold is crossed. The notification is not generated as soon as the threshold is crossed. The notification with jnxMbgPgwSMAlarmState = “RAISED” is generated if this notification has not already been generated for a threshold or the trap with jnxMbgPgwSMAlarmState = “CLEARED” has been generated for a threshold and the number stays above a threshold for a duration of 3 minutes (default) . The notification with jnxMbgPgwSMAlarmState = “CLEARED” is generated if the notification with jnxMbgPgwSMAlarmState = “CLEARED” has been generated for a threshold and the number stays below the threshold for a duration of 3 minutes (default). jnxMbgPgwSMAlarmThrshld indicates the threshold that was crossed. jnxMbgPgwSMAlarmState (RAISED/CLREAD) indicates if the number is more than the threshold (“RAISED”) or is less than the threshold (“CLEARED”).
  - jnxMbgPgwSMSubscribersThresPerSPNotif

    The total number of subscribers per services PIC has crossed a threshold. jnxMbgPgwSMSPICName indicates the services PIC for which this trap was generated.
  - jnxMbgPgwSMSessionEstFailThresPerSPNotif

    The total number of session establishment failures per Service PIC has crossed a threshold. jnxMbgPgwSMSPICName indicates the services PIC for which this trap was generated.
  - jnxMbgPgwSMSessionEstFailThresPerTCNotif

    The total number of session establishment failures per traffic class (GTPv1) has crossed a threshold. jnxMbgPgwSMQCIName indicates the TC (Traffic Class) for which this trap was generated.
  - jnxMbgPgwSMSessionEstFailThresPerQCINotif

    The total number of session establishment failures per QoS class identifier (GTPv2) has crossed a threshold. jnxMbgPgwSMQCIName indicates the QCI for which this trap was generated.

- jnxMbgPgwSMBearersThresGblNotif

  The total number of bearers in the system has crossed a threshold.

- jnxMbgPgwSMBearersThresPerSPNotif

  The total number of bearers per services PIC has crossed a threshold. jnxMbgPgwSMSPICName indicates the services PIC for which this trap was generated.

Related Documentation

- Troubleshooting Mobility on page 15

Troubleshooting Admission Control

This topic discusses class of service (CoS) and call admission control (CAC) serviceability.

To troubleshoot call admission control, you should understand the classifier policy profiles configured on your system. A classifier policy is the configuration that maps QCI (4G) and TC/THP (3G) to internal forwarding queues and defines packet loss priority. You can have multiple classifier policy profiles on your system. Therefore, understanding how these multiple classifiers interact with your system and with each other is key to understanding what to look for when you have problems with admission control.

To understand CoS, you must understand the CoS policy. This policy is the configuration that manges quality of service (QoS) parameters. You can have multiple CoS policies on your system.

CoS and CAC serviceability also depends on two other configurations:

- Resource threshold policy which controls your system for CAC. You can have multiple resource threshold policies configured on your system.
- The bandwidth pool, which allocates bandwidth sharing among APNs and the gateway. You can have multiple bandwidth pools configured on your system.

Finally, you need to know about local policies. A local policy is a collection of a classifier profile, a CoS policy profile, a resource threshold policy profile, and a bandwidth pool. A local policy is so termed because it is attached to the gateway or to individual APNs.

You can troubleshoot class of service and call admission control by examining:

- Total system bandwidth and per APN bandwidth can be configured with percentage allocations to each QCI/Traffic-Class.
- System ensures each QCI gets allocated system bandwidth optimally
- Maximum-bearers configuration for the gateway
- High or low threshold percentages for CPU, memory, system load, or maximum bearers with local policy to apply when a threshold is reached
- Forwarding-class or loss-priority definition per QCI or traffic class
- Local policy to cap maximum GBR, MBR, and AMBR values per APN

Use the following commands to troubleshoot this environment:

- For subscribers, use the command:

  user@host > **show unified-edge ggsn-pgw subscribers extensive**

- For preemption lists (priority levels), use the command:

  user@host > **show unified-edge ggsn-pgw status preemption-list detail**

To debug QoS negotiation parameters:

1. Check the session status to determine whether it is a visitor, roaming, or home session.
2. Look up the local policy being applied to the APN.
3. Match this local policy with its classifier profile, the CoS policy, and the bandwidth pool

To troubleshoot calls rejected by CAC:

1. Identify rejected calls by entering:

   user@host > **show unified-edge ggsn-pgw qos statistics apn** ***apn-name***l

   Counters such as “No resources”, “Service denied”, “Authentication Fail”, “APN access denied” indicate rejected calls, but not necessarily by CAC.

2. To verify the cause for rejected calls, look in the Routing Engine stats section:

   Active Bearers
   CPU Load (%)
   Memory Load (%)

   These counters can indicate that the system is running out of resources.

3. To verify that resource exhaustion is the source of the problem, enter these commands:

   user@host > **show unified-edge rmps table gateway-bearers**
   user@host > **show unified-edge rmps table apn-bearers**
   user@host > **show unified-edge rmps table anchor-pfe-bandwidth**
   user@host > **show unified-edge rmps table bandwidth-pools**

Related Documentation

- Troubleshooting Mobility on page 15

Monitoring AAA Metrics

AAA server metrics include:

- Server Up/Down status traps
- Network element status traps
- Real-time latency and flow control statistics

RADIUS logs are useful for troubleshooting an AAA profile. The following sections show logs for create, update, delete, and dynamic requests.

Create Session requests communicate with the S-GW, the P-GW, and the RADIUS server in the following manner:

S-GW --> Create Session request --> P-GW --> Access Request --> RADIUS
P-GW <-- Access Accept <-- RADIUS
P-GW --> Accounting Start request ->> RADIUS
S-GW <-- Create Session response <-- P-GW
P-GW <-- Accounting Start response <-- RADIUS
S-GW <-- Create Session response <-- P-GW

If **apn wait-accounting** is enabled (it is disabled by default), then the P-GW sends the Create Session response after receiving the Accounting Start response.

The following RADIUS logs show how these Create Session requests are processed. When there is a breakdown in AAA communications, the logs help you isolate the cause of the problem.

1. Access the RADIUS log, enable trace options, and look for Authentication and Accounting messages.

```
Jun 24 11:50:19 1001025 gtid:[26]tid: [2] jsimRadius(2) Access-Request
IP 10.10.2.11 20024 >

...

Jun 24 11:50:19 1013620 gtid:[24]tid: [0] Access-Accept

...

Jun 24 11:50:19 1022764 gtid:[25]tid: [1] jsimRadius(1)
Accounting-Request IP 10.10.2.11 20025 >

...

Jun 24 11:50:19 1033840 gtid:[26]tid: [2] Accounting-Response
```

Interim requests can be configured to generate accounting requests periodically or they are generated when the S-GW generates a Modify bearer Request. When a Modify bearer Request is received, communication with the S-GW, P-GW, and RADIUS server flows in the following manner:

S-GW --> Modify bearer request --> P-GW
P-GW --> Interim request --> RADIUS
P-GW <-- Dynamic request <-- RADIUS
P-GW <-- CoA <-- RADIUS
S-GW <-- Update bearer request <-- P-GW
S-GW --> Update bearer response --> P-GW
P-GW ---> CoA ACK --> RADIUS
P-GW ---> Interim accounting response --> RADIUS

NOTE: Modify bearer requests are generated by subscriber location information changes, QoS changes, roaming, time-zone changes, and so on.

The following RADIUS logs show how these interim Accounting messages are processed. When there is a breakdown in AAA communications, the logs help you isolate the cause of the problem.

1. Access the RADIUS log, enable trace options, and look for Authentication and Accounting messages.

```
Jun 24 11:58:28 879452 gtid:[25]tid: [1] Accounting-Request

...

Jun 24 11:58:28 880542 gtid:[25]tid: [1] Acct-Status-Type [40] 4 0000
0003

...

Jun 24 11:58:28 880818 gtid:[25]tid: [1] Acct-Input-Octets [42] 4 0000
00064 <- Data flow

...

Jun 24 11:58:28 880849 gtid:[25]tid: [1] Acct-Output-Octets [43] 4 0000
00064 <- Data flow

...

Jun 24 11:58:28 891299 gtid:[25]tid: [1] Accounting-Response
```

Accounting stop (delete) requests communicate with the S-GW, the P-GW, and the RADIUS server in the following manner:

S-GW --> Delete Session request --> P-GW
P-GW --> Accounting Stop request --> RADIUS
S-GW <-- Delete Session response <-- P-GW
S-GW <-- Delete Session request <-- P-GW
P-GW --> Accounting Sstop --> RADIUS

For a dynamic stop request, the flow is:

P-GW <-- Disconnect request <-- RADIUS
S-GW <-- Delete Session request <-- P-GW
P-GW --> Accounting Stop --> RADIUS

The following RADIUS logs show how these Delete Accounting messages are processed. When there is a breakdown in AAA communications, the logs help you isolate the cause of the problem.

1. Access the RADIUS log, enable trace options, and look for Accounting Stop messages.

```
Jun 24 12:06:29 957706 gtid:[25]tid: [1] jsimRadius(1) Accounting-Request
IP 10.10.2.11 20025 >

...

Jun 24 12:06:29 958502 gtid:[25]tid: [1] Acct-Status-Type [40] 4 0000
0002

...

Jun 24 12:06:29 958785 gtid:[25]tid: [1] Acct-Input-Octets [42] 4 0000
00c8 <- Data flow

...

Jun 24 12:06:29 958815 gtid:[25]tid: [1] Acct-Output-Octets [43] 4 0000
00c8 <- Data flow

...

Jun 24 12:06:29 974810 gtid:[26]tid: [2] Accounting-Response
```

NOTE: In the displays in this section, **acct-status-type** ends with a four-digit code. The last number of this code is meaningful. A code that ends in **0001** means the process is starting. A code that ends in **0002** means the process is stopping. A code that ends in **0003** means the process is in an interim state, which allows parameters to be changed.

The following aggregate show commands are useful for troubleshooting AAA processes.

- To show AAA statistics authentication details for a specific interface:

  user@host> **show unifed-edge ggsn-pgw aaa statistics authentication detail fpc-slot 3 pic-slot 0**

- To show AAA statistics accounting details for a specific interface:

  user@host> **show unifed-edge ggsn-pgw aaa statistics accounting detail fpc-slot 3 pic-slot 0**

- To show AAA statistics authentication details for a specific PIC:

  user@host> **show unifed-edge ggsn-pgw aaa statistics authentication detail**

- To show AAA statistics accounting details for a specific PIC:

  user@host> **show unifed-edge ggsn-pgw aaa statistics accounting detail**

- To show AAA statistics accounting details for a specific RADIUS server interface:

  user@host> **show unifed-edge ggsn-pgw aaa statistics radius authentication detail fpc-slot 3 pic-slot 0 name jsimRadius**

- To show AAA statistics accounting details for a specific RADIUS server interface:

  user@host> **show unifed-edge ggsn-pgw aaa radius statistics accounting detail fpc-slot 3 pic-slot 0 name jsimRadius**

- To show network element status lists of the RADIUS servers and their status:

  user@host> **show unified-edge ggsn-pgw aaa network-element status name ne1 fpc-slot 3 pic-slot 0**

  ```
  Network-element: ne1
    Server: radius1, Priority: 1, State: Active
    Server: radius2, Priority: 1, State: Active
    Server: radius3, Priority: 2, State: Active
  ```

The following clear commands are useful for detecting ongoing activity:

- To clear AAA authentication statistics:

  user@host> **clear unifed-edge ggsn-pgw aaa statistics authentication**

- To clear AAA accounting statistics:

  user@host> **clear unifed-edge ggsn-pgw aaa statistics accounting**

- To clear RADIUS server authentication statistics:

  user@host> **clear unifed-edge ggsn-pgw aaa radius statistics authentication**

- To clear RADIUS server accounting statistics:

  user@host> **clear unifed-edge ggsn-pgw aaa radius statistics accounting**

The following test commands are useful for debugging problems:

- To test user authentication:

  user@host> **test unified-edge ggsn-pgw aaa authentication fpc-slot 1 pic-slot 0 profile abc charging-id 0xffffff username aaa password aaa**

- To start an accounting test:

```
user@host> test unified-edge ggsn-pgw aaa accounting fpc-slot 1 pic-slot 0 profile
  abc charging-id 0xffffff start
```

- To stop the accounting test:

```
user@host> test unified-edge ggsn-pgw aaa accounting fpc-slot 1 pic-slot 0 profile
  abc charging-id 0xffffff stop
```

- To test the interim interval configuration:

```
user@host>test unified-edge ggsn-pgw aaa accounting fpc-slot 1 pic-slot 0 profile abc
  charging-id 0xffffff interim
```

PART 4

Index

Index

S

T