

cnSGW-C Troubleshooting

- Description, on page 1
- Using CLI Data, on page 1
- Logs, on page 8

Description

This chapter provides information on using the command line interface (CLI) commands and logs for troubleshooting any issues that may arise during system operation.

Using CLI Data

This section describes the show and clear commands and the monitor commands that are used for troubleshooting

show subscriber and cdl show Commands

Table 1: Feature History

Feature Name	Release Information	Description
Enhanced session management and clearing options	2026.02.0	

Feature Name	Release Information	Description
		This feature introduces new filtering parameters to the existing show and clear subscriber commands. These enhancements allow operators to target sessions based on their activity state (idle or active) or the specific duration the session has been connected. Additionally, a max-count limit can be applied to restrict the number of sessions processed or displayed, providing better control over system resources during bulk operations. **Commands introduced:** The following options are added to the show subscriber, show subscriber count, and clear subscriber commands: • **sub-state**: Filters or clears sessions based on the subscriber's current activity state. • **active**: Targets sessions currently in an active state. • **idle**: Targets sessions currently in an idle state. • **connected-duration**: Filters or clears sessions based on the length of time the session has been connected. • • **greater-than <seconds>**: Targets sessions connected longer than the specified value. • **lesser-than <seconds>**: Targets sessions connected for less than the specified value. • **max-count**: Limits the number of subscriber sessions

Feature Name	Release Information	Description
		returned or cleared. • **lesser-than <value>**: Specifies the maximum number of sessions to be impacted by the command.
Supporting IPv6 Only eNB Insertion through Show and Clear Subscriber CLI commands	2024.02.0	Before you add IPv6 only eNBs in a network, all UPFs in a mesh must be IPv6 enabled for successful handovers of IPv4 only eNB sessions to IPv6 only eNB sessions. In addition, all sessions must have V4V6 tunnel before inserting a V6 only eNB. To support this IPV6 only eNB insertion, cnSGW-c includes the following CLI commands: • The **show subscriber nf-service sgw data-tunnel** *data_tunnel_type* and **show subscriber count nf-service sgw data-tunnel** *data_tunnel_type* CLI commands • The **clear subscriber nf-service sgw data-tunnel** *data_tunnel_type* CLI command **Default Setting**: Not Applicable

This section describes troubleshooting information.

- To display the SGW subscriber information, use the following commands:

  show subscriber namespace sgw imsi *imsi_value*

  show subscriber nf-service sgw imsi *imsi_value*

  show subscriber count { all }

```
show subscriber namespace sgw imsi 123456789012348
subscriber-details
{
  "subResponses": [
    {
      "status": true,
      "genericInfo": {
        "imsi": "imsi-123456789012348",
        "mei": "imei-123456786666660",
```

```
  "msisdn": "msisdn-223310101010101",
  "accessType": "EUTRAN",
  "plmnId": {
    "mcc": "123",
    "mnc": "456"
  },
  "sgwProfileName": "sgw1",
  "unAuthenticatedImsi": "No"
},
"s11cInterfaceInfo": {
  "sgwTeid": "[0x12000147] 301990215",
  "sgwIPv4Address": "209.165.201.19",
  "mmeTeid": "[0x62b5] 25269",
  "mmeIPv4Address": "209.165.201.20"
},
"pdnInfoList": {
  "totalPdn": 1,
  "pdnInfo": [
    {
      "pdnId": "PDN-1",
      "apn": "intershat",
      "attachType": "Initial Attach",
      "sgwRelocState": "N/A",
      "operatorPolicyName": "N/A",
      "dnnProfileName": "N/A",
      "defaultEbi": 5,
      "pdnType": "IPv4",
      "allocatedIPv4": "209.165.201.26",
      "apnSelectionMode": "Subscribed",
      "ambrUplink": "10 Kbps",
      "ambrDownlink": "20 Kbps",
      "s5cInterfaceInfo": {
        "sgwTeid": "[0x52000147] 1375732039",
        "sgwIPv4Address": "209.165.201.19",
        "pgwTeid": "[0x339a] 13210",
        "pgwIPv4Address": "209.165.201.18"
      },
      "sxaInterfaceInfo": {
        "selectedUP": "209.165.201.20",
        "upEpKey": "209.165.201.20:209.165.201.19",
        "cpSeid": "[0x1200014752000147] 1297038098512740679",
        "upSeid": "[0x2712] 10002"
      },
      "bearerInfoList": {
        "totalBearer": 1,
        "bearerInfo": [
          {
            "bearerId": "Bearer-1",
            "state": "Connected",
            "ebi": 5,
            "isDefaultBearer": true,
            "qosInfo": {
              "qci": 6,
              "arp": 113
            },
            "s1uInterfaceInfo": {
              "sgwTeid": "[0x62b7] 25271",
              "sgwIPv4Address": "209.165.200.226",
              "eNodeBTeid": "[0x62b8] 25272",
              "eNodeBIPv4Address": "209.165.201.20"
            },
            "s5uInterfaceInfo": {
              "sgwTeid": "[0x62b6] 25270",
              "sgwIPv4Address": "209.165.201.1",
```

```
                    "pgwTeid": "[0x339b] 13211",
                    "pgwIPv4Address": "209.165.201.18"
                  },
                  "chargingId": 303174163
                }
              ]
            },
            "uli": {
              "mcc": "123",
              "mnc": "456",
              "tac": "0x92a",
              "eci": "0x12d687"
            },
            "uetimeZone": {
              "timeZone": "+0:15",
              "dayLightSavingTime": "+1 hour"
            },
            "plmnType": "VISITOR"
            "dualStackEnabled": true,
            "dataTunnelType": "IPV4"
            "ipSrcViolation": "N/A",
            "mmeS4SgsnId": {
              "IPv4": "10.10.10.1"
            }
          }
        ]
      }
    }
  ]
}
```


```
show subscriber count all
subscriber-details
{
   "sessionCount": 50
}
```

- To display the session summary information, use the following command:

  cdl show sessions summary

```
cdl show sessions summary
message params: {session-summary cli session {0 100 0 [] 0 0}}
session {
    primary-key imsi-146062234105885
    unique-key [ 16777218 ]
    map-id 1
    instance-id 1
    version 1
    create-time 2020-04-27 16:18:24.225646626 +0000 UTC
    last-updated-time 2020-04-27 16:18:24.87241245 +0000 UTC
    purge-on-eval false
    next-eval-time 2020-05-04 16:18:24 +0000 UTC
    data-size 406
}
```

- To display the subscriber output based on the data tunnel type, use the following command:

  show subscriber nf-service sgw data-tunnel *data_tunnel_type*

```
show subscriber nf-service sgw data-tunnel IPV4V6
subscriber-details
{
  "subResponses": [
    [
      "id-index:1:0:32768",
```

```
      "id-index-key:1:0:globalKey:32768",
      "id-value:16777344",
      "imsi:imsi-123456789012348",
      "msisdn:msisdn-223310101010101",
      "imei:imei-123456786666660",
      "upf:10.1.12.212",
      "upfEpKey:10.1.12.212:10.1.10.78",
      "subscribertype:non-volte",
      "s5s8Ipv4:10.1.15.100",
      "s11Ipv4:10.1.12.212",
      "data-tunnel:IPV4V6",
      "namespace:sgw",
      "nf-service:sgw"
    ]
  ]
}
```

- To display the subscriber count output based on data tunnel type, use the following command:

  show subscriber count nf-service sgw data-tunnel *data_tunnel_type*

```
show subscriber count nf-service sgw data-tunnel IPV4V6
subscriber-details
{
  "sessionCount": 1
}
```

- To clear subscriber information, use the following commands:

  clear subscriber all

  clear subscriber nf-service sgw all

```
clear subscriber all
result
ClearSubscriber Request submitted
```

```
clear subscriber nf-service sgw all
result
ClearSubscriber Request submitted
```

- To clear the subscriber information using a data tunnel type, use the following command:

  clear subscriber nf-service sgw data-tunnel *data_tunnel_type*

```
clear subscriber nf-service sgw data-tunnel IPV4V6
result
Clear subscriber request submitted successfully for GR Instance ID 1. Waiting Time is
9.351706 seconds
```

Enhanced session management and clearing options

New filtering parameters are introduced to the existing **show subscriber**, **show subscriber count**, and **clear subscriber** CLI commands to enhance session management and clearing options in cnSGW.

The following options are added to the **show subscriber**, **show subscriber count**, and **clear subscriber** commands.

- **sub-state**: Filters or clears sessions based on the subscriber's current activity state.
  - **active**: Targets sessions currently in an active state.
  - **idle**: Targets sessions currently in an idle state.

Example:

```
show subscriber namespace sgw gr-instance 1 sub-state idle
show subscriber namespace sgw gr-instance 1 sub-state active
clear subscriber namespace sgw gr-instance 1 sub-state idle
clear subscriber namespace sgw gr-instance 1 sub-state active
show subscriber count namespace sgw gr-instance 1 sub-state idle
show subscriber count namespace sgw gr-instance 1 sub-state active
```

- **connected-duration**: Filters or clears sessions based on the length of time the session has been connected.
  - **greater-than <seconds>**: Targets sessions connected longer than the specified value.
  - **lesser-than <seconds>**: Targets sessions connected for less than the specified value.

Example:

```
show subscriber namespace sgw gr-instance 1 connected-duration greater-than <seconds>
show subscriber namespace sgw gr-instance 1 connected-duration lesser-than <seconds>
clear subscriber namespace sgw gr-instance 1 connected-duration greater-than <seconds>
clear subscriber namespace sgw gr-instance 1 connected-duration lesser-than <seconds>
show subscriber count namespace sgw gr-instance 1 connected-duration greater-than
<seconds>
show subscriber count namespace sgw gr-instance 1 connected-duration lesser-than <seconds>
```

- **max-count**: Limits the number of subscriber sessions returned or cleared.
  - **lesser-than <value>**: Specifies the maximum number of sessions to be impacted by the command.

Example:

```
show subscriber namespace sgw max-count { lesser-than <count> }
clear subscriber namespace sgw max-count { lesser-than <count> }
```

Logs

The system logging feature provides a common way to log the log messages across applications. Each log consists of the following components:

- Timestamp—Shows the date and time of the log creation.
- Log message—Shows the message of a specific log.
- Log level—Shows the level of importance of log message.
- Log tag—Shows the details of module name, component name, and interface name. A log tag is pre-created and passes during logging.

Logs for Event Failures

Table 2: Feature History

Feature Name	Release Information	Description
Event Failure Logs for Service Pods	2025.03.0	With this feature, the consistent event failure logs are enhanced to support the IDFT, LI, Suspend Notification, Resume Notification, and PDN Disconnect (GTPU Path Failure) procedures for the service pods.
Event Failure Logs for Service Pods	2025.01.0	With this feature, the consistent event failure logs are enhanced to support the Delete Bearer Command, Change Notification, and PDN Disconnect procedures for the service pods.
Event Failure Logs for Service Pods	2024.04.0	With this feature, the consistent event failure logs are enhanced to support the Create Bearer, Update Bearer, Delete Bearer, PDN Modify List, and Modify Bearer Command procedures for the service pods.

Feature Name	Release Information	Description
Event Failure Logs	2024.03.0	cnSGWc provides the following support: • Consistent event failure logs for PDN Setup, Idle or Active, PDN Modify, and PDN Disconnect procedures across pods • Configurable logs at pod type • Inclusion of request and response details in a single-line format The significant volume of unnecessary system-generated logs resulted in increased memory consumption, performance impact, and ineffective management and utilization of logs. To prevent these issues, the consistent error log message format across various pods is introduced for reduced memory consumption, minimized number of log generations by the system, and efficient troubleshooting. The single-line log format display enhances the readability. The significant volume of unnecessary system-generated logs resulted in increased memory consumption, performance impact, and ineffective management and utilization of logs. To prevent these issues, the consistent error log message format across various pods is introduced. The enhanced error logging for cnSGWc procedures provides significant improvements, such as reduced memory consumption, minimized number of log generations by the system, detailed, consistent, and configurable logging that help in effective debugging and system monitoring. **Default Setting:** Not Applicable

The error logging capabilities for cnSGWc procedures are enhanced for providing detailed, consistent, and configurable logging. These enhancements help in effective debugging and system monitoring. These enhancements are:

- Consistent log format—The single-line log format is standardized across different pods, such as REST endpoint and service pods, to ensure uniformity in how logs are recorded and interpreted.
- Enabling and disabling logging—An option to enable or disable logging is available at specific pod types. This option provides flexibility in managing log storage.
- Detailed log content—Logs include comprehensive details, such as primary key, interface, procedure details, message requests, and responses. This level of detail helps in thorough debugging and analysis.
- Log level management—By default, logs are written at the INFO level. You can enable the logs, as required. The log enablement helps in controlling the amount of log data generated and stored.

  By default, the log level is set to WARN, which ensures that logging is disabled by default. You can enable the logging, as required.
- Common logging interface—A common interface is implemented for event logging. All components use this interface for the same logging standards and formats.
- Log tags—Log tags are enhanced to allow you to enable or disable logs for specific pod types or services and provides granular control over logging.
- Supported logs—Logging is supported for various procedures, pods, and interfaces.

How it Works

To have the consistent log format across each pod, each component uses a common interface for event logging. A log uses the JSON format so that all the data appears in a single line. Logs are written at INFO level so that this level can be disabled by default and enabled, as required.

A log tag has the following format:

transaction.event.<pod-type>, where **<pod-type>** is the service name that a pod uses. For example, rest-ep and sgw-service.

You can enable or disable logs for a specific pod type or service using log tags.

The logs are written when the corresponding log level is set to INFO, DEBUG, or TRACE. However, the message request or response fields are populated only when the log level is set to DEBUG or TRACE.

Sample Log when Debug level is enabled

```
sgw-service-0 [INFO] [Transaction.go:1595] [transaction.event.sgw-service]
 {"TxnId":16,"StartTime":"2024-05-10T15:01:00+05:30","GRInstanceId":1,
"TxnType":"S5CreateSessReq","Priority":33,"SessionNamespace":"sgw(2)",
"CdlSliceName":"1","SubscriberId":"imsi-430967582185910","SessionPrimaryKey":
"imsi-430967582185910","SessionKeys":"imsi-430967582185910 (pk)
subscribertype:wps (nuk) 16777217 (uk) id-index:1:0:32768 (nuk)
id-index-key:1:0:globalKey:32768 (nuk) id-value:16777217 (nuk) imsi:
imsi-430967582185910 (nuk) msisdn:msisdn-9326737733 (nuk) imei:imeisv-1122334455667788
 (nuk) upf:192.168.56.20 (nuk) upfEpKey:192.168.56.20:192.168.56.10
(nuk) s11Ipv4:192.168.56.20 (nuk) s5s8Ipv4:192.168.56.30 (nuk)",
"SessionState":"Create_Session","ErrorMessage":"{\"ErrType\":3,\"ErrCause\":
{\"Value\":89,\"Pce\":false,\"Bce\":false,\"OrigInd\":false,\"OffendingIe\":
{\"Valid\":false,\"Tag\":0,\"Instance\":0,\"Value\":\"\"}},\"BrCtxtCause\":true,
\"SubfailReason\":89,\"SubfailReasonDetailed\":0,\"SubfailStr\":\
"S5 Create Session Response Failure\",\"SubfailReasonStr\":\"IPv4:192.168.56.30
```

```
IPv6:\",\"SourceDetails\":\"/opt/workspace/sgw_service/src/sgw-service/procedures
/pdnsetup/idlestate.go:963\"}","MessageRequest":{"Version":2,"TEIDflag":true,
"MsgPriority":true,"MsgTypeId":32,"MsgPriorityValue":10,"meta_data":{"from_ip"
:3232249867,"to_ip":3232249886,"to_port":2123,"intfType":3,"s5edscp":{"Value":12,
"Valid":true},"peerType":1},"MsgType":{"Create_Session_Request":{"IMSI":
"430967582185910","APN":"starent.com","AMBR":{"UL":119,"DL":135},"MEI":
"1122334455667788","MSISDN":"9326737733","Indication":{},"PAA":{"PDN_Type":1,
"IPv4":"0.0.0.0"},"RAT_Type":{"value":6},"Serving_Network":{"MCC":"123","MNC":
"765"},"ULI":{"uli_tai":{"mcc":"214","mnc":"365","val":4660}},"FQ_TEID":{"sgwCntrl":
{"IFace":6,"TEID":1358954497,"IPv4":"192.168.56.11"}},"Bearer_Context_List":
{"num_bearer_ctxt":1,"pbBearerCxt":[{"linked_ebi":{"value":5},"fqteid":{"sgwData":
{"TEID":1073807935,"IPv4":"192.168.56.60","IPv6":"4123::192:168:56:60"}},
"bearerQos":{"PL":2,"QCI":5,"arp":8}}]},"Charging_Characteristics":{"value":
"NBIAAA=="},"PDN_Type":{"value":1},"APN_Restriction":{},"Selection_Mode":
{"value":1}}}},"MessageResponse":{"Version":2,"TEIDflag":true,"MsgPriority":
true,"MsgLength":19,"TIED":1358954497,"Seq":1,"MsgTypeId":33,"MsgPriorityValue":10,
"meta_data":{"from_ip":3232249886,"to_ip":3232249867,"intfType":3},"MsgType":
{"Create_Session_Response":{"Cause":{"Cause_Value":89},"Recovery":{"value":10}}}}}
```


Sample Log when Info level is enabled


```
sgw-service-0 [INFO] [Transaction.go:1607] [transaction.event.sgw-service]
 {"TxnId":7,"StartTime":"2024-06-20T12:10:05+05:30","GRInstanceId":1,"TxnType":
"S5CreateSessReq","Priority":33,"SessionNamespace":"sgw(2)","CdlSliceName":"1",
"SubscriberId":"imsi-430967582185910","SessionPrimaryKey":"imsi-430967582185910",
"SessionKeys":"imsi-430967582185910 (pk) subscribertype:wps (nuk) 16777217 (uk)
id-index:1:0:32768 (nuk) id-index-key:1:0:globalKey:32768 (nuk) id-value:16777217
(nuk) imsi:imsi-430967582185910 (nuk) msisdn:msisdn-9326737733 (nuk) imei:
imeisv-1122334455667788 (nuk) upf:192.168.56.20 (nuk)
upfEpKey:192.168.56.20:192.168.56.10 (nuk) s11Ipv4:192.168.56.20 (nuk)
s5s8Ipv4:192.168.56.30 (nuk)","SessionState":"Create_Session","ErrorMessage":
"{\"ErrType\":3,\"ErrCause\":{\"Value\":89,\"Pce\":false,\"Bce\":false,\"OrigInd\"
:false,\"OffendingIe\":{\"Valid\":false,\"Tag\":0,\"Instance\":0,\"Value\":\"\"}},
\"BrCtxtCause\":true,\"SubfailReason\":89,\"SubfailReasonDetailed\":0,\"SubfailStr\":
\"S5 Create Session Response Failure\",\"SubfailReasonStr\":\"IPv4:192.168.56.30
IPv6:\",\"SourceDetails\":\"/opt/workspace/sgw_service/src/sgw-service/procedures
/pdnsetup/idlestate.go:971\"}"}
```


Supported Logs

This feature supports logs for cnSGWc to provide detailed error information across various procedures, pods, and interfaces. Following table lists the supported logs.

Table 3: Supported Logs

cnSGWc Procedures with Supported Logging	Pod Involved in Logging	cnSGWc Interfaces Associated with Pod and Procedure
PDN Setup	SERVICE	S11, S5, SXA, RMGR
Idle/Active	SERVICE	S11, S5, SXA
PDN Modify	SERVICE	S11, SXA, S5
PDN Disconnect (DSR/DBR)	SERVICE	S11, S5, SXA, RMGR
Create Bearer	SERVICE	S11, S5, SXA
Update Bearer	SERVICE	S11, S5, SXA
Delete Bearer (Dedicated)	SERVICE	S11, S5, SXA

cnSGWc Procedures with Supported Logging	Pod Involved in Logging	cnSGWc Interfaces Associated with Pod and Procedure
PDN Modify List	SERVICE	S11, S5, SXA
Modify Bearer Command	SERVICE	S11, S5, SXA
Delete Bearer Command	SERVICE	S11, S5, SXA
Change Notification	SERVICE	S11, S5, SXA
PDN Disconnect	SERVICE	S11, S5, SXA, RMGR
IDFT	SERVICE	S11, SXA
LI	SERVICE	X2, X3
Suspend Notification	SERVICE	S11, S5, SXA
Resume Notification	SERVICE	S11, S5, SXA
PDN Disconnect (GTPU Path Failure)	SERVICE	S11, S5, SXA, RMGR

Enable Event Logging

Use the following procedure to enable or disable the event logs of the cnSGWc service. The appropriate log level configuration using the CLI command allows you to control the amount and type of log data generated. Hence, this configuration helps in effective monitoring, troubleshooting, and performance management.

Procedure

Step 1 Specify the log level for event logging.

logging name transaction.event.<*podname*> **level application** *application_log_level*

The <pod-type> is the service name that a pod uses.

Example:

```
 logging name transaction.event.sgw-service level application [ debug | error | info | off | trace | warn ]
```

Step 2 [Optional] Use **show running-config logging** command to verify the configured application event logging level.

Example:

```
show running-config logging
   logging level tracing debug
   logging name infra.config.core level application trace
   logging name infra.config.core level transaction trace
   logging name infra.config.core level tracing off
   logging name infra.message_log.core level transaction trace
   logging name transaction.event.sgw-service level application debug
```