GNMI-Telemetry Plugin

Table of contents

The GNMI Telemetry Plugin is a server that uses the gNMI protocol to stream data from UFM telemetry. Users can select the data to stream, specify intervals, and choose to include only deltas (on-change mode).

The server supports three functions: Capability, Get and Subscribe.

Data Streaming: The streamed data is delivered in CSV format. Headers are provided in the first message and included in subsequent messages. Data is presented in hex format to conserve space for unchanged data. Values are displayed as an array of strings, each representing a unique identifier (GUID) and port. Depending on the mode, values may have missing rows if there are no changes in the GUID and port.

Metadata Streaming: The plugin can stream UFM's metadata, providing an inventory of it. For convenience, examples use the gNMIc client, but any gNMI client can be used.

Configuration and Polling Intervals: The polling intervals for each server cache are configurable with the following defaults:

- Telemetry: every 5 minutes
- Inventory: every minute
- Events: every minute
- Switch rank: every 6 hours

The service supports telemetry from switch-level data (fset) and port-level data (xcset), querying `low_freq_debug` xcset by default. Multiple telemetries can be polled simultaneously.

Data Sharding: The service supports sharding the cache data on request, allowing many clients to request the same data while each receiving a different part.

Deployment

To deploy the plugin with UFM (SA or HA):

1. Install the latest version of UFM.
2. Run UFM with `/etc/init.d/ufmd start`.

3. Pull the plugin image from the Docker Hub.

4. Run `/opt/ufm/scripts/manage_ufm_plugins.sh add -p gnmi_telemetry -t <version>` to enable the plugin, or use the UFM UI to add the plugin via Setting   Plugin Management   Right Click on GNMI-telemetry   Add   select version   Add.

5. Check that the plugin is running with `docker ps`.

6. If the gNMI default port is unavailable, change the config file `gnmi_telemetry.env` and restart the plugin.

Authentication

The server's authentication is determined by the gNMI protocol. Two configurable items require authentication: the UFM Telemetry URL and the UFM inventory IP.

- Authentication is not necessary for the UFM telemetry URL. Therefore, only the telemetry URL is required.
- The inventory is sourced from the UFM of the local host, but can be changed to a different machine in the config file. To do so, token access to that machine is necessary.

Secure Server

The server can be secured using certificates. To secure the server, set the "secure_mode_enabled" flag to "true" in the configuration (default is true). Upon initialization, the gNMI server retrieves UFM certificates from the `/opt/ufm/conf/webclient` folder. The certificate folder can be changed by modifying the shared volume.

The server requires client-call certificates, granting access only if client certificates match its own. The gNMI server periodically checks its certificates for updates, ensuring they remain up-to-date. The client certification naming convention must align with the DNS name (SAN) as the UFM.

Supported API Requests

The service supports the following requests:

- **Capability**: Describes the YANG files the service supports (UFM telemetry).

- **Get**: Requires legal paths; receives the cache data from the service.
- **Subscribe**: Requires legal paths and an interval; receives cache data at the specified interval. The first message contains headers extracted from the path, and subsequent messages include only the headersID. In on-change subscribe mode, a heartbeat interval is provided instead of an interval. During the heartbeat interval, if no data changes, no notification is sent; A full notification message, similar to the first message, is sent. If some data changes a notification of the change is sent; No heart message is send.

Capability Request

The capability request provides information about the YANG files that the server supports, including their versions. This request can be fulfilled without requiring a connection to the telemetry or inventory.

Example:

```
gnmic -a localhost:9339 capability
```

Example Response:

```
gNMI version: 1.2.11
supported models:
  - nvidia-ib-amber, Nvidia IB, 1.0.0
  - nvidia-ib-amber-ext, Nvidia IB, 1.0.0
  - nvidia-ib-amber-inventory-counters, Nvidia IB, 1.0.0
  - nvidia-ib-amber-port-counters, Nvidia IB, 1.0.0
supported encodings:
  - JSON
  - JSON_IETF
```

Supported Paths

Telemetry Request Path Construction

To construct a path for a telemetry request, follow these steps:

1. Begin with "nvidia/ib".
2. Specify sharding if desired. For example, to partition the data into 10 pieces and take the second partition, use 2/10.
3. Specify the node_guid to select, using an asterisk (*) to select all nodes.
4. Specify the desired ports for the selected nodes, using an asterisk (*) to select all ports.
5. Select "amber" for amBER telemetry.
6. Specify the desired counters group. If unknown, this step can be skipped.
7. Specify the counter, using an asterisk (*) to select all the counters in the cache. If a counters group is used, it will return all counters in the specified group.

Other Information Requests (Events, Inventory)

1. Begin with "nvidia/ib".
2. Specify inventory or events.

Switch Rank Information Path Construction

To construct a path for switch rank information, follow these steps:

1. Begin with "nvidia/ib".
2. Specify the node_guid to select, using an asterisk (*) to select all nodes.
3. Select "amber" for amBER telemetry.
4. Use Switch_rank as the counter name.

Additional Configurations

Name	Description	Default
grpc_port	The port that the service uses for the gNMI protocol	9339
cpu_list	The CPU cores that the server is allowed to use	3
include_port_sharding	Includes the ports in the sharding. Ports of the node may not be in the same shard	false
filtered_columns	Specifies which counters to ignore when querying the telemetry	port_guid
ufm_ip	If querying events and inventory from a different UFM instance, provide the IP of that host machine and a UFM token for remote UFM	127.0.0.1
ufm_access_token	Required if the ufm_ip is not local	

Telemetry Messages - Data Format

Telemetry messages consist of two key components: Headers and Values, both representing telemetry data in a CSV format.

- **Headers**: Initially provided in a full mode, but transition to a string hash format after the second message when using a subscribe request to reduce message size.

- **Values**: Each value begins with a timestamp, followed by the node_guid and port number, and then the counter value in the same order as the headers. If a counter is not present for a node, it will be empty in the message.

In on-change subscribe messages, only nodes with changes and their corresponding modified values are included. All other counters for that node will remain empty.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/hist0 --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/hist1 -i 30s
[{ "source": "localhost:9339",
  "subscription-name": "default-1690282472",
  "timestamp": 1690282475124352063,
  "time": "2023-07-25T13:54:35.124352063+03:00",
  "updates": [{ "Path": "nvidia/ib/amber/reply/sample", "values": { "nvidia/ib/amber/reply/sample": {
          "Headers": "timestamp,guid,port,hist0,hist1",
                 "HeaderID": "5246201354",
           "Values": ["240771222771818,0x8168793592c6a790,1,,2",
            "240771222771818,0x47a67159c915493f,1,1,2",
...
            "240771222771818,0x667203ac69f3f2bf,1,2,",
            "240771222771818,0x113cd807bfed3853,1,0,"
          ]}}}]}
```

The second message on the headers will be set to hash values.

Get Request

The Get request retrieves data at a specified path. If the telemetry is devoid of information, the server will respond with an empty response. Otherwise, it will respond with counters it can locate.

Example:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_counters/hist0
```

The request retrieves data from node_guid `0x5255456`, specifically in port number 2, with the request counter set to hist0.

Example 2:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/hist0
```

The request retrieves the data from all the ports and the node_guids, with the request counter set to hist0.

Example3:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/*
```

The request retrieves the data from node_guid `0x5255456`, port 2, with the request counters set to "all".

Example for multi path:

```
gnmic -a localhost:9339 --insecure get
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/CableInfo.transmitter_technology --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/sel_gctrln_en_5_lane0 --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/num_plls_7nm --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/rcal_fsm_done --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/LinkErrorRecoveryCounterExtended --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/sel_enc2_ib0_lane2 --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/lockdet_err_cnt_unlocked_sticky
```

Response Example:

```
[{
  "source": "localhost:9339",
  "timestamp": 1719232374915165166,
  "time": "2024-06-24T15:32:54.915165166+03:00",
  "updates": [{
    "Path": "nvidia/ib/amber/reply",
    "values": {
     "nvidia/ib/amber/reply": {
          "Headers":
["timestamp","Node_GUID","Port_Number","CableInfo.transmitter_technology","sel_gctrln_en_5_lane0","n
          "Values": [
            "1719232345757948,0x91f87bf42deb3e03,1,5091,7826,6290,8615,4247,8586,6214",
            "1719232345757948,0x7b8c2e08907250ce,1,2891,3293,5774,4398,3681,3548,7408",
            "1719232345757948,0x48b60e6f3670eaca,1,9477,3847,1184,5527,4783,2102,8192",
            "1719232345757948,0xabccdad7f8a3eda6,1,7976,6143,8257,3770,6166,6690,2835",
            "1719232345757948,0x6d9ec4bb5fa45736,1,9051,2982,7145,3604,9256,1061,2638",
            "1719232345757948,0x028cf9e0f9ed7c32,1,5623,7483,2263,2265,6890,4875,5564",
            "1719232345757948,0x92a984c1a491b72a,1,6732,7795,6411,8569,3370,705,5536",
            "1719232345757948,0x8b4b404acd2f34da,1,7610,7128,10064,1880,4834,3411,6724",
            "1719232345757948,0x20f92ed58991d56c,1,6805,1632,5407,2038,1865,7279,8350",
            "1719232345757948,0x1dac004a426bb5f5,1,8351,5757,7925,6181,3260,3081,1554"
          ]
        }}}]}]
```

Subscribe Stream Request

The Subscribe request, similar to the get request, provides data from the specified path. When the telemetry is empty, the server responds with an empty result. If data is available, the server responds with the retrieved counters. The stream delivers information at the specified interval. If no interval is specified, the server transmits the information at the default server rate, which is configurable and defaults to 10s.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_counters/hist0 -i 30s
```

This request retrieves data from the node_guid `0x5255456`, port 2, where the request counter is hist0, and the interval is configured for 30 seconds. If the user wishes to test the stream, the stream mode can be configured to "once," and following a single response, the stream will be stopped.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
```

```
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_counters/hist0 -i 30s --mode once
```

This request retrieves the data from node_guid `0x5255456`, port 2, where the request counter is hist0. The stream shuts down after one response, similar to a Get request.

Example:

```
gnmic -a localhost:9339 --insecure sub --path nvidia/ib/guid[guid=*]/port[port_number=*]/amber/* -i
10s
```

The server responds for the first two notifications, as follows:

```
{
  "source": "localhost:9339",
  "subscription-name": "default-1719233128",
  "timestamp": 1719233128171946518,
  "time": "2024-06-24T15:45:28.171946518+03:00",
  "updates": [{
     "Path": "nvidia/ib/amber/reply/sample",
     "values": {
       "nvidia/ib/amber/reply/sample": {
         "HeaderID": "970426048",
         "Headers":
["timestamp","Node_GUID","Port_Number","Counter1","Counter2","Counter3","Counter4","Counter5","Co
"Counter7"],
         "Values": [
           "1719232345757948,0x91f87bf42deb3e03,1,5091,7826,6290,8615,4247,8586,6214",
           "1719232345757948,0x7b8c2e08907250ce,1,2891,3293,5774,4398,3681,3548,7408",
           "1719232345757948,0x1dac004a426bb5f5,1,8351,5757,7925,6181,3260,3081,1554",
           "1719232345757948,0x48b60e6f3670eaca,1,9477,3847,1184,5527,4783,2102,8192",
           "1719232345757948,0xabccdad7f8a3eda6,1,7976,6143,8257,3770,6166,6690,2835",
           "1719232345757948,0x6d9ec4bb5fa45736,1,9051,2982,7145,3604,9256,1061,2638",
           "1719232345757948,0x028cf9e0f9ed7c32,1,5623,7483,2263,2265,6890,4875,5564",
           "1719232345757948,0x92a984c1a491b72a,1,6732,7795,6411,8569,3370,705,5536",
           "1719232345757948,0x8b4b404acd2f34da,1,7610,7128,10064,1880,4834,3411,6724",
           "1719232345757948,0x20f92ed58991d56c,1,6805,1632,5407,2038,1865,7279,8350"
         ]}}}]}

{  "source": "localhost:9339",
  "subscription-name": "default-1719233128",
  "timestamp": 1719233138173907825,
  "time": "2024-06-24T15:45:38.173907825+03:00",
  "updates": [{
     "Path": "nvidia/ib/amber/reply/sample",
     "values": {
       "nvidia/ib/amber/reply/sample": {
         "HeaderID": "970426048",
         "Values": [
           "1719232345757948,0x20f92ed58991d56c,1,6805,1632,5407,2038,1865,7279,8350",
           "1719232345757948,0x1dac004a426bb5f5,1,8351,5757,7925,6181,3260,3081,1554",
           "1719232345757948,0x48b60e6f3670eaca,1,9477,3847,1184,5527,4783,2102,8192",
           "1719232345757948,0xabccdad7f8a3eda6,1,7976,6143,8257,3770,6166,6690,2835",
           "1719232345757948,0x6d9ec4bb5fa45736,1,9051,2982,7145,3604,9256,1061,2638",
           "1719232345757948,0x028cf9e0f9ed7c32,1,5623,7483,2263,2265,6890,4875,5564",
           "1719232345757948,0x92a984c1a491b72a,1,6732,7795,6411,8569,3370,705,5536",
           "1719232345757948,0x8b4b404acd2f34da,1,7610,7128,10064,1880,4834,3411,6724",
           "1719232345757948,0x91f87bf42deb3e03,1,5091,7826,6290,8615,4247,8586,6214",
           "1719232345757948,0x7b8c2e08907250ce,1,2891,3293,5774,4398,3681,3548,7408"
         ]}}}]}
```

Subscribe On-Change Request

The subscribe on-change request, similar to the standard subscribe request, provides data from the specified path. If the telemetry lacks data, the server responds with an empty result. When data is available, the server responds with the located counters. The stream delivers information at the specified interval, but only if there is new information to transmit. Otherwise, it waits for the next interval to check for updates. The path construction follows the same pattern as the get request and includes inventory and event paths.

Only updated data will be included in the response, with all other parts remaining empty but retaining the specified format. Similarly, only the nodes that have been updated will be included in the response.

Example:

```
gnmic -a localhost:9339 --insecure  sub --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_counters/hist0  --stream-mode on-
```

```
change --heartbeat-interval 1m
```

This request retrieves data from node_guid `0x5255456`, port 2, with the request counters set to hist0. It periodically checks for changes every minute, and when changes are detected, it promptly sends the updated values.

Example:

```
gnmic -a localhost:9339 --insecure  sub --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/* --stream-mode on-change --
heartbeat-interval 1m
```

This request involves all nodes and ports, aiming to retrieve all counters from the telemetry. It periodically checks for changes every minute, and when changes are detected, it promptly sends the updated values.

The below is an example of the response to a particular GUID, which represents an on-change request for a few counters. However, only specific counters have been updated, those who have not updated have a value of 0. Because the flag `include_old_data_on_change`default is true

```
1706532307824,0x0002c903007e5220,1,0,0,0,41447490564,617155163,41423305825,617155163,2418473
```

The same example with the flag set to true will give this:

```
1706532307824,0x0002c903007e5220,1,,,,41447490564,617155163,41423305825,617155163,24184739,1
```

Only the values that have changed return while the others are empty values. To get this format of data, one need to change the `include_old_data_on_change` in the config file to false.

Example:

```
gnmic -a localhost:9339 --insecure sub --path nvidia/ib/guid[guid=*]/port[port_number=*]/amber/* --
```

```
stream-mode on-change --heartbeat-interval 24h
```

The server responds for the first 2 notifications are the following (where include_old_data_on_change is true), one can see the last two columns have not changed but still return the data before, the second message was send due to some rows have changed, those rows

```
{
 "source": "localhost:9339",
 "subscription-name": "default-1719236764",
 "timestamp": 1719236764654659517,
 "time": "2024-06-24T16:46:04.654659517+03:00",
 "updates": [{
    "Path": "nvidia/ib/amber/reply/onchange",
    "values": {
      "nvidia/ib/amber/reply/onchange": {
        "HeaderID": "912200528",
        "Headers":
["timestamp","Node_GUID","Port_Number","Counter1","Counter2","Counter3","Counter4","Counter5","Co
        "Values": [
          "1719236753818594,0x7e680fb8f81a1950,1,100531,107250,100999,107455,109258,3716,5329",
          "1719236753818594,0x0176438fe4ee507c,1,104269,108884,104887,108502,105366,4540,6673",
          "1719236753818594,0x2e36224302959e79,1,101228,100555,105616,102767,108899,87,9953",

"1719236753818594,0x8e62a55d7571a9b8,1,100684,108124,106670,102400,106689,2910,4203",
          "1719236753818594,0x0be75a9e97016f5e,1,102227,102735,108903,103547,108705,2629,1830",
          "1719236753818594,0x8307bfad0672adbd,1,106033,103906,106185,107450,105736,2567,6914",
          "1719236753818594,0x2cbe66ec0b1af84c,1,105958,106959,100349,107704,105073,8330,4962",
          "1719236753818594,0x6b6da39a9ec4bbfc,1,104340,106752,109134,103796,103500,7136,3493",
          "1719236753818594,0x6d122dbdd99cfb60,1,104941,107630,104190,105392,109582,5480,7934",
          "1719236753818594,0xeed4bd9cd3b7f325,1,102416,100164,106731,102033,103807,3048,6316"

]}}}]}



{
 "source": "localhost:9339",
 "subscription-name": "default-1719236764",
 "timestamp": 1719237054620929561,
 "time": "2024-06-24T16:50:54.620929561+03:00",
 "updates": [
  {
    "Path": "nvidia/ib/amber/reply/onchange",
    "values": {
      "nvidia/ib/amber/reply/onchange": {
        "HeaderID": "912200528",
        "Values": [
          "1719237054172043,0xeed4bd9cd3b7f325,1,117416,115164,121731,117033,118807,3048,6316",
          "1719237054172043,0x2e36224302959e79,1,116228,115555,120616,117767,123899,87,9953",

"1719237054172043,0x8e62a55d7571a9b8,1,115684,123124,121670,117400,121689,2910,4203",
          "1719237054172043,0x7e680fb8f81a1950,1,115531,122250,115999,122455,124258,3716,5329",
          "1719237054172043,0x0176438fe4ee507c,1,119269,123884,119887,123502,120366,4540,6673"
        ]}}}]}
```

Inventory Requests

Inventory messages are conveyed in separate updates, presenting the inventory details of the UFM associated with the provided IP. These messages display comprehensive information, including the total count of various components within the UFM, such as switches, routers, servers, and more, along with details about active ports and the total number of ports, including disabled ones. Moreover, inventory requests include the size of the telemetry, which is not always the same as the active ports. In cases where the plugin is unable to establish contact with the UFM, it will revert to using default values defined in the configuration file. It is worth noting that the path for inventory requests differs from the conventional path structure, as they do not rely on specific nodes or ports. Consequently, inventory requests are initiated after "nvidia/ib."

Example:

```
gnmic -a localhost:9339 --insecure get –path nvidia/ib/inventory/*
```

Response:

```
[{
    "source": "localhost:9339",
    "timestamp": 1698824237536878067,
    "time": "2023-11-01T09:37:17.536878067+02:00",
    "updates": [{
        "Path": "nvidia/ib/inventory",
        "values": {
          "nvidia/ib/inventory": {
            "ActivePorts": 4, "Cables": 2, "Gateways": 0, "HCAs": 2, "Routers": 0,”Servers": 2, "Switches": 1,
"TotalPorts": 38, "TelemetrySize": 4, "timestamp": 1698824211535069000}
     }}]}]
```

Events Requests

Events messages are provided in separate updates, offering insights into the events occurring within the UFM associated with the specified IP. Given that the event metadata remains consistent, even when numerous events are part of a request, the message format adopts a CSV-like structure. The Headers section contains essential metadata regarding UFM events, while the Values section contains the raw event data. Users can subscribe to these events with the on-change feature enabled, receiving only the events triggered within the subscription interval. Notably, the path structure for event requests differs from the typical node or port-based structure and is requested after "nvidia/ib."

Example:

```
gnmic -a localhost:9339 --insecure get –path nvidia/ib/events/*
```

Response:

```
[ {
    "source": "localhost:9339",
    "timestamp": 1698824809647515575,
    "time": "2023-11-01T09:46:49.647515575+02:00",
    "updates": [  {
        "Path": "nvidia/ib/events",
        "values": {
          "nvidia/ib/events": {
            "Headers": [
"id","object_name","write_to_syslog","description","type","event_type","severity","timestamp","counter","ca
            "Values": [
              "7718,Grid,false,Disk space usage in /opt/ufm/files/log is above the threshold of
90.0%.,Grid,525,Critical,2023-11-01 07:25:54,N/A,Maintenance,Grid,Disk utilization threshold reached",
          "7717,Grid,false,Disk space usage in /opt/ufm/files/log is above the threshold of 
90.0%.,Grid,525,Critical,2023-11-01 07:24:54,N/A,Maintenance,Grid,Disk utilization threshold reached",
          "7716,Grid,false,Disk space usage in /opt/ufm/files/log is above the threshold of 
90.0%.,Grid,525,Critical,2023-11-01 07:23:54,N/A,Maintenance,Grid,Disk utilization threshold reached",
....
          "7491,ec0d9a0300d42e54,false,Mcast group is deleted: ff12601bffff0000,
00000002,Computer,67,Info,2023-10-31 06:39:21,N/A,Fabric Notification,default / Computer: r-
ufm59,MCast Group Deleted"]
    }}}]}]
```

Switch Rank Requests

Switch rank updates are conveyed in separate messages, presenting the rank of the switches in the UFM. This data is derived from a file in the UFM and is updated by the server every 6 hours by default. The `switch_rank` counter is associated only with switch-level data, so there is no need to specify a port in the path. However, this counter is not connected to the telemetry cache of switch-level data. **Note that if the `ufm_ip` is changed, the switch_rank information will not be available.**

Example:

```
gnmic -a localhost:9339 --insecure get --path nvidia/ib/guid[guid=*]/amber/switch_rank

Respond for example:



 {
   "source": "localhost:9339",
   "timestamp": 1719296207323383222,
   "time": "2024-06-25T09:16:47.323383222+03:00",
   "updates": [
     {
       "Path": "nvidia/ib/guid[guid=*]/amber/amber/switch_rank",
       "values": {
         "nvidia/ib/guid/amber/amber/switch_rank": {
           "Headers": "Timestamp,Node_GUID,switch_rank",
           "Values": [
             "1719296205612,0x0002c903007e5220,0"
```

```
        ]}}}]}
```