GNMI-Telemetry Plugin

Table of contents

Authentication

Secure Server

Capability Request

Get Request

Subscribe Stream Request

Subscribe On-Change Request

Messages Data Format

Inventory Requests

Events Requests

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

The GNMI server is designed to support four functions: capability, get, subscribe, and set. However, it should be noted that the server does not currently support the "set" function, only "capability," "get," and "subscribe."

The streamed data is delivered in CSV format. Headers are initially provided in the first message, and subsequently, they are included in every other message. The data is presented in hex format to conserve space for data that remains unchanged. The values are presented as an array of strings, each representing a unique identifier (GUID) and port.

Depending on the selected mode, the values may have missing rows if there have been no changes in the GUID and port.

Furthermore, the plugin has the capability to stream UFM's metadata by providing an inventory of it. While the provided examples will use the gNMIc client for convenience, this functionality can work with any gNMI client.

Authentication

The server's authentication is determined by the gNMI protocol, and whether it is secured or unsecured is specified in the configuration. Two configurable items require authentication: the UFM Telemetry URL and the UFM inventory IP. Both of these items must be configured in the configuration file.

- Authentication is not necessary for the UFM telemetry URL. Therefore, only the telemetry URL is required.
- By default, the inventory is sourced from the UFM of the local host. However, changing the UFM inventory location to a different machine in the config file is possible. To do so, token access to that machine is necessary.

Secure Server

The server can be secured by using certificates. To secure the server, the "secure_mode_enabled" flag need to be set to "true" in the configuration, on default, the server security is configured to be true.

Upon initialization, the gNMI server retrieves the UFM certificates from the /opt/ufm/conf/webclient folder, utilizing both the server certificates and CA certificates. It is possible to change the certificate folder by changing the shared volume.

The server requires certificates for client calls and grants access only if the client certificates match its own. The gNMI server periodically examines its certificates for updates and ensures that they remain up to date. In addition, the server requires that the client certifications naming convention is aligned with the DNS name (SAN) as the UFM.

Capability Request

Description: The capability request provides information about the Yang files that the server supports, including their versions. This request can be fulfilled without the need for a connection to the telemetry or inventory.

Example:

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

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.

The path construction follows these steps:

1. Begin with "nvidia/ib"
2. Specify the node_guid that the user wants to select, with an asterisk (*) representing a selection of all nodes.
3. Choose the desired ports for the selected nodes.
4. Select "amber" and the desired counters group, and then specify the counter.

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".

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. However, if there is data available, the server responds with the counters it can locate. The stream delivers information at intervals corresponding to the requested interval. If a user fails to specify an interval, the server will transmit the information as soon as it becomes available. The path construction follows the same pattern as the get request.

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.

Subscribe On-Change Request

The subscribe on-change request, much like the standard subscribe request, provides data from the specified path. In the event that the telemetry lacks data, the server responds with an empty result. However, when data is available, the server responds with the counters it can locate. The stream delivers information according to the interval specified in the request, but only if there is new information to transmit. Otherwise, it will wait for the next interval to check the telemetry for updates. The path construction follows the same pattern as the get request. On-change requests contain inventory and event paths as well.

Importantly, only the data that has been updated will be included in the response; all other parts will be empty but retain 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_changedefault is true

```
1706532307824,0x0002c903007e5220,1,0,0,0,41447490564,617155163,41423305825,617155163,24184
```

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

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

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

Messages Data Format

Telemetry messages consist of two key components: Headers and Values, both representing the telemetry data in CSV format. When utilizing a subscribe request, the headers transition to a string hash format after the second message, primarily to conserve message size. In the case of on-change subscribe messages, there is an additional adjustment where only nodes that have undergone changes are included, along with their corresponding modified values. All other counters for that node will remain empty.

Each value within the "Values" section starts with a timestamp, followed by the node_guid and port number, and then the value of the counter, maintaining the same order as the headers. If a specific counter is not present for the node, it will remain empty in the message.

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",
        "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.

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 includes 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","
          "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"]       }
      } }   ] }]
```