Probleemoplossing voor API-gebaseerde EPNM-meldingen

Inhoud

Inleiding

Dit document beschrijft hoe u EPNM-meldingen kunt oplossen wanneer REST API wordt gebruikt om toegang te krijgen tot foutinformatie van het apparaat.

Voorwaarden

De client die u implementeert, moet geschikt zijn om de twee mechanismen die door Evolved Programmable Network Manager (EPNM) worden gebruikt, te verwerken en te abonneren om meldingen te verzenden.

Vereisten

Er zijn geen specifieke vereisten van toepassing op dit document.

Gebruikte componenten

Dit document is niet beperkt tot specifieke software- en hardware-versies.

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

EPNM API-meldingen

Kennisgevingen waarschuwen netwerkbeheerders en exploitanten over belangrijke gebeurtenissen of

problemen die verband houden met het netwerk. Deze meldingen helpen ervoor te zorgen dat potentiële problemen snel worden gedetecteerd en opgelost, waardoor de downtime wordt beperkt en de algehele netwerkprestaties worden verbeterd.

EPNM kan verschillende methoden verwerken, zoals meldingen via e-mail, Simple Network Management Protocol (SNMP)-traps naar specifieke ontvangers of Syslog-berichten naar externe Syslog-servers. Naast deze methoden biedt EPNM ook een Representational State Transfer Application Programming Interface (REST API) die kan worden gebruikt om informatie op te halen over inventaris, alarmen, servicesactivering, sjabloonuitvoering en hoge beschikbaarheid.

Op API gebaseerde meldingen worden momenteel ondersteund met het gebruik van twee verschillende mechanismen:

- **Verbindingsgerichte meldingen:**De client abonneert op een vooraf gedefinieerde URL en gebruikt een WebSocket-client met basisverificatie via een beveiligd HTTPS-kanaal.
- **Berichten zonder verbinding:** de gebruiker wordt verwacht een REST webservice te hebben die in staat is om de payloads van extensible markup language (XML) en/of JavaScript Object Notation (JSON) te accepteren als een POST-verzoek.

Alle meldingen delen hetzelfde schema en kunnen worden opgehaald in JSON of XML-formaten.

Basis EPNM-configuratie

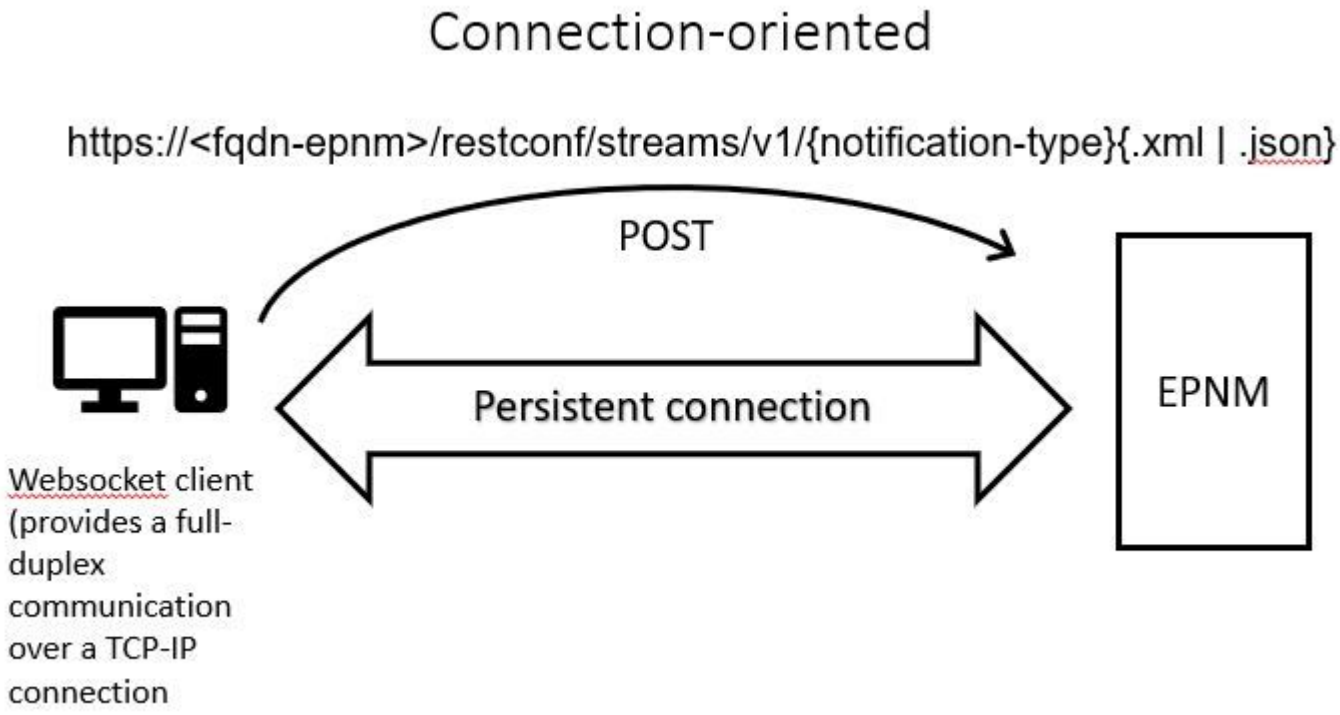
Standaard worden alarm- en inventarismeldingen uitgeschakeld. Om deze in te schakelen, wijzigt u de restconf-config.properties bestand zoals aangegeven (de EPNM-toepassing hoeft niet opnieuw te worden gestart):

```
/opt/CSCOlumos/conf/restconf/restconf-config.properties

epnm.restconf.inventory.notifications.enabled=true
epnm.restconf.alarm.notifications.enabled=true
```

Verbindingsgerichte meldingen

In het beeld, de cliëntmachine stelt een WebSocket in werking en abonneert aan EPNM met een vooraf bepaalde URL, met basisauthentificatie, en door een veilig kanaal HTTPS.



Een WebSockets Python-client uitvoeren

De WebSocket-client bibliotheek in Python kan gebruikt worden om een WebSocket te maken in de client machine.

```
import websocket
import time
import ssl
import base64

def on_message(ws, message):
    print(message)

def on_error(ws, error):
    print(error)

def on_close(ws, close_status_code, close_msg):
    print("### closed \###")

def on_open(ws):
    ws.send("Hello, Server!")

if __name__ == "__main__":
    username = "username"
    password = "password"
    credentials = base64.b64encode(f"{username}:{password}".encode("utf-8")).decode("utf-8")
    headers = {"Authorization": f"Basic {credentials}"}
    websocket.enableTrace(True)
    ws = websocket.WebSocketApp("wss://10.122.28.3/restconf/streams/v1/inventory.json",
                                on_message=on_message,
                                on_error=on_error,
                                on_close=on_close,
                                header=headers)
```

```
    ws.on_open = on_open
    ws.run_forever(sslopt={"cert_reqs": ssl.CERT_NONE})
```

Abonnement op een verbindingsgeoriënteerde client

Met deze code wordt een WebSocket-client ingesteld die zich abonneert op EPNM op **wss://10.122.28.3/restconf/streams/v1/inventory.json**. Het gebruikt de Python WebSocketbibliotheek om de verbinding tot stand te brengen en berichten in en uit te verwerken. Het abonnement kan ook zijn (op basis van welk soort bericht je wilt abonneren):

/restconf/streams/v1/alarm{.xml | .json}

/restconf/streams/v1/service-activation{.xml | .json}

/restconf/streams/v1/template-execution{.xml | .json}

/restconf/streams/v1/all{.xml | .json}

Het on_message, on_error en on_close functies zijn callback functies die worden opgeroepen wanneer de WebSocket verbinding een bericht ontvangt, een fout tegenkomt, of wordt gesloten, respectievelijk. Het **on_open** De functie is een callback die wordt opgeroepen wanneer de WebSocket verbinding wordt gevestigd en klaar om te gebruiken.

Het username en password De variabelen worden ingesteld op de aanmeldingsgegevens die nodig zijn voor de toegang tot de externe server. Deze referenties worden vervolgens gecodeerd met de base64 toegevoegd aan de kopregels van het WebSocket-verzoek.

Het run_forever De methode wordt opgeroepen op het object WebSocket om de verbinding te starten en het voor onbepaalde tijd open te houden en te luisteren naar berichten die van de server komen. Het sslopt De parameter wordt gebruikt om de SSL/TLS-opties voor de verbinding te configureren. Het CERT_NONE vlaggen maakt certificatie-validering onmogelijk.

Voer de code uit om de WebSocket klaar te hebben om de meldingen te ontvangen:

```
(env) devasc@labvm:~/epnm$ python conn-oriented.py
--- request header ---
GET /restconf/streams/v1/inventory.json HTTP/1.1
Upgrade: websocket
Host: 10.122.28.3
Origin: https://10.122.28.3
Sec-WebSocket-Key: shY1K9SqXTphBqaZFh/iMQ==
Sec-WebSocket-Version: 13
Connection: Upgrade
Authorization: Basic cm9vdDpQYXNzMTIzNA==


-----------------------
--- response header ---
HTTP/1.1 101
Set-Cookie: JSESSIONID=5BFB68B0126226A0A13ABE595DC63AC9; Path=/restconf; Secure; HttpOnly
Strict-Transport-Security: max-age=31536000;includeSubDomains
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Upgrade: websocket
Connection: upgrade
```

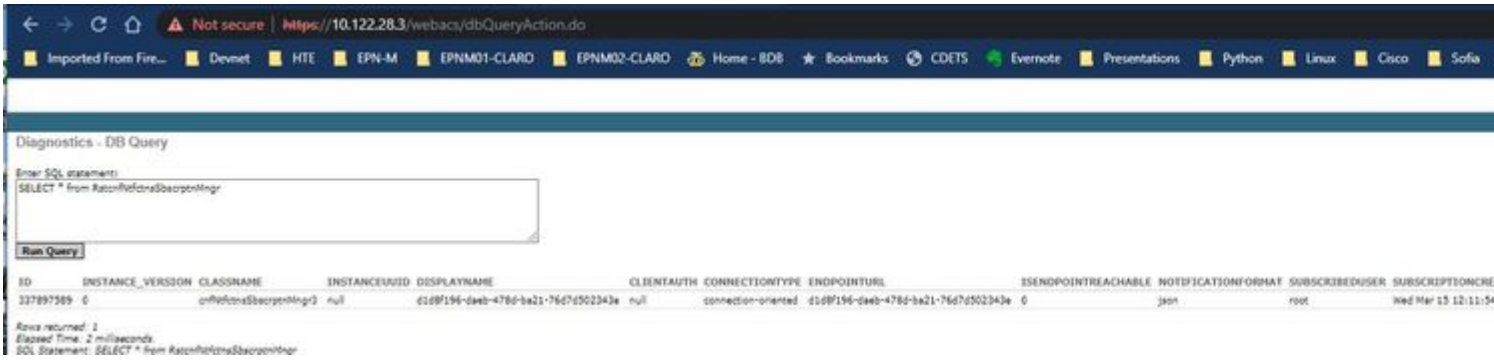
```
Sec-WebSocket-Accept: Ozns7PGgHjrXj0nAgnlhbyVKPjc=
Date: Thu, 30 Mar 2023 16:18:19 GMT
Server: Prime
-----------------------
Websocket connected
++Sent raw: b'\x81\x8es\x99ry;\xfc\x1e\x15\x1c\xb5R*\x16\xeb\x04\x1c\x01\xb8'
++Sent decoded: fin=1 opcode=1 data=b'Hello, Server!'
++Rcv raw: b'\x81\x0eHello, Server!'
++Rcv decoded: fin=1 opcode=1 data=b'Hello, Server!'
Hello, Server!
```

U kunt de kennisgevingsabonnementen op de server controleren met deze DB-query (navigeer naar https://

/webacs/dbQueryAction.do

).

```
SELECT * from RstcnfNtfctnsSbscrptnMngr;
```



Uit de online EPNM-documentatie, zodra deze tot stand is gebracht, wordt dezelfde verbinding in stand gehouden gedurende de hele levenscyclus van de toepassing:

- totdat de client de verbinding met de server verbreekt
- tot de server uitvalt voor onderhoud of tijdens een failover

Als u om de een of andere reden een bepaald abonnement moet verwijderen, kunt u een HTTP DELETE verzoek bij de SUBSCRIPTIONID in de URL gespecificeerd https://

. Voorbeeld:

```
devasc@labvm:~/epnm$ curl --location --insecure --request DELETE 'https://10.122.28.3/restconf/data/v1/c
> --header 'Accept: application/json' \
> --header 'Content-Type: application-json' \
> --header 'Authorization: Basic cm9vdDpQYXNzMTIzNA==' \
> --header 'Cookie: JSESSIONID=2CB4F43E3D4BCE5F42411114065F6292'
```

Verificatie van berichten, DEBUG-vermeldingen; show log**, gebruikte bestandsnaam, SQL-uitgangen**

Om uit te zoeken waarom een client die een verbindingsgeoriënteerd mechanisme gebruikt meldingen niet goed ontvangt, kunt u de aangegeven DB-query uitvoeren en controleren of het abonnement al dan niet aanwezig is. Als het niet aanwezig is, vraag de klant eigenaar om ervoor te zorgen om het abonnement uit te geven.

In de tussentijd kunt u DEBUG-niveau inschakelen in com.cisco.nms.nbi.epnm.restconf.notifications.handler.NotificationsHandlerAdapter u kunt dit dus opvangen wanneer het abonnement wordt verzonden:

```
[root@epnm-spo-lab-host SCRIPTS]# sudo /opt/CSCOlumos/bin/setLogLevel.sh com.cisco.nms.nbi.epnm.restconf
Loglevel set to DEBUG for com.cisco.nms.nbi.epnm.restconf.notifications.handler.NotificationsHandlerAdap
```

Nadat het abonnement is verzonden, kunt u controleren of een vermelding met het IP-adres van de WebSocket-client wordt weergegeven in localhost_access_log.txt:

```
[root@epnm-spo-lab-host SCRIPTS]# zgrep -h '"GET /restconf/streams/.* HTTP/1.1" 101' $(ls -1t /opt/CSCO
```

```
10.134.4.35 - - [30/Mar/2023:15:33:43 -0300] "GET /restconf/streams/v1/all.json HTTP/1.1" 101 -
```

Controleer tot slot nogmaals de OB (merk op dat de tijdstempel overeenkomt met de ingang in localhost_access_log.txt.

Het volgende logboek toont wanneer de POST verzoeken om abonnementen worden verzonden:

```
[root@epnm-spo-lab-host SCRIPTS]# grep -Eh 'DEBUG com.cisco.nms.nbi.epnm.restconf.notifications.handler.
```

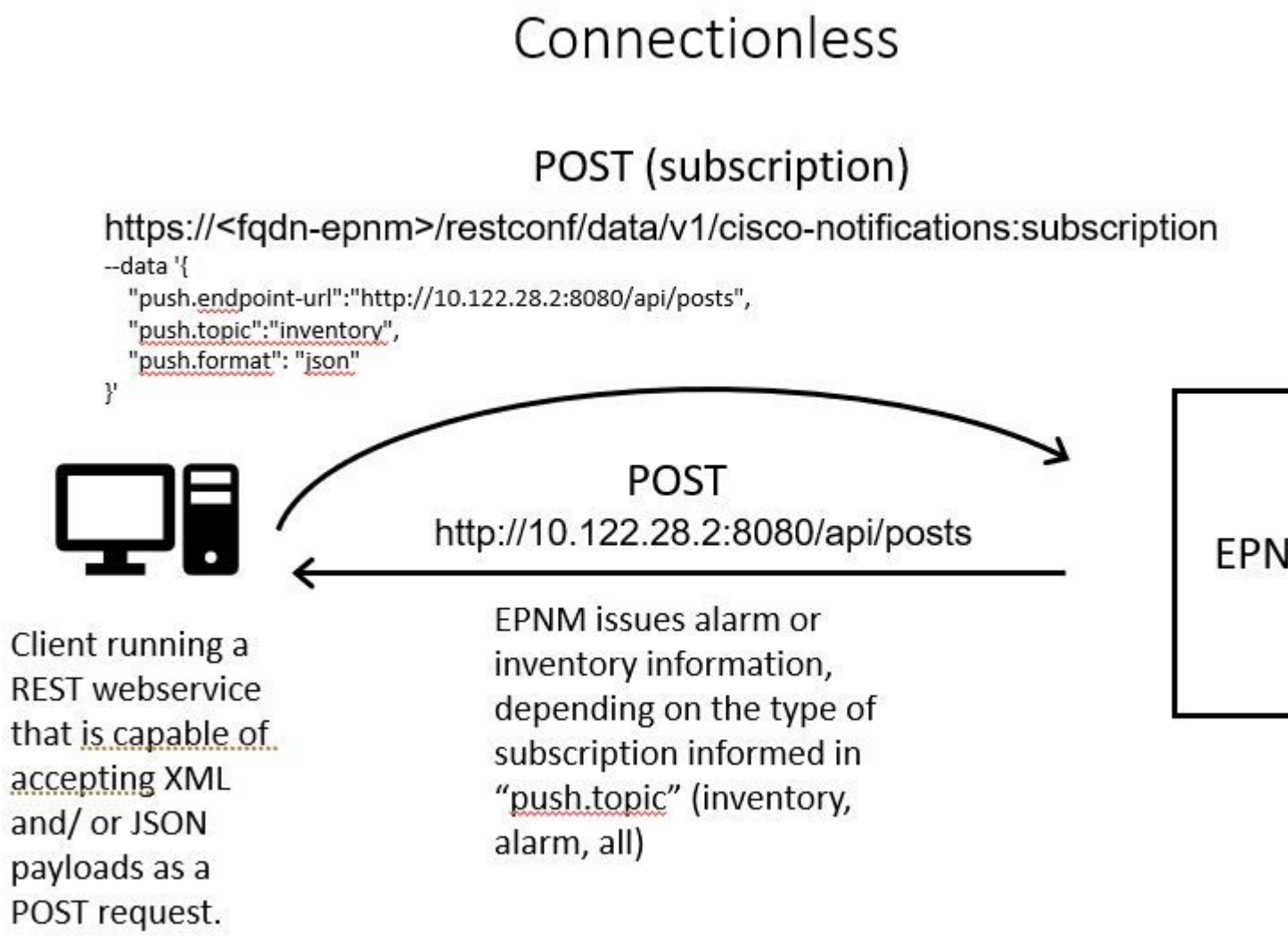
```
2023-03-30 15:33:43,399: DEBUG com.cisco.nms.nbi.epnm.restconf.notifications.handler.Notificatior
```

Zolang de verbinding in leven wordt gehouden, wordt een bericht van type push-change-update verzonden van de EPN-M server naar alle clients die zich hebben geabonneerd op meldingen. Het voorbeeld laat een van de meldingen zien die door het EPNM worden verstuurd wanneer de hostnaam van een NCS2k wordt gewijzigd:


```
{
  "push.push-change-update":{
    "push.notification-id":2052931975556780123,
    "push.topic":"inventory",
    "push.time-of-update":"2023-03-31 13:50:36.608",
    "push.time-of-update-iso8601":"2023-03-31T13:50:39.681-03:00",
    "push.operation":"push:modify",
    "push.update-data":{
      "nd.node":{
        "nd.description":"SOFTWARE=ONS,IPADDR=10.10.1.222,IPMASK=255.255.255.0,DEFRTR=255.255.255.255,IP
        "nd.equipment-list":"",
        "nd.fdn":"MD=CISCO_EPNM!ND=tcc222c",
        "nd.sys-up-time":"217 days, 14:40:170.00"
      }
    }
  }
}
```


Meldingen zonder verbinding

Het volgende is de workflow in het geval van connectionless kennisgevingen:



Een REST Webservice Python-client uitvoeren

De gebruiker wordt verwacht een REST webservice te hebben die in staat is om XML en/of JSON payloads te accepteren als een POST-verzoek. Deze REST-service is het eindpunt waar de Cisco EPNMhet kader voor herstellingmeldingen publiceert meldingen. Dit is een Voorbeeld van een REST webservice die in de afstandsbediening moet worden geïnstalleerd:

```
from flask import Flask, request, jsonify

app = Flask(__name__)

@ app.route('/api/posts', methods=['POST'])
def create_post():
    post_data = request.get_json()
    response = {'message': 'Post created successfully'}
    print(post_data)
    return jsonify(response), 201

if __name__ == '__main__':
    app.run(debug=True, host='10.122.28.2', port=8080)
```

Dit is een Python Flask webapplicatie die één eindpunt definieert /api/posts die aanvaardt **HTTP POST** verzoeken. Het create_post() functie wordt aangeroepen wanneer een **HTTP POST** verzoek wordt ingediend bij /api/posts. In de create_post() de gegevens van het verzoek dat wordt ingediend, worden opgehaald met het gebruik van **request.get_json()**, die een woordenboek van de lading van JSON terugkeert. De payload wordt dan afgedrukt met **print(post_data)** voor debug-doeleinden. Daarna wordt een antwoordbericht aangemaakt met de toets message en waarde **Post created successfully** (in woordenboekformaat). Dit antwoordbericht wordt vervolgens teruggestuurd naar de client met een HTTP-statuscode van 201 (gemaakt).

Het if __name__ == '__main__': Block is een standaard Python constructie die controleert of het script als hoofdprogramma draait, in plaats van geïmporteerd als een module. Als het script wordt uitgevoerd als het hoofdprogramma, wordt de Flask-applicatie gestart en wordt deze uitgevoerd op het opgegeven IP-adres en de poort. Het **debug=True** Het argument maakt debug-modus mogelijk, die voorziet in gedetailleerde foutmeldingen en het automatisch opnieuw laden van de server wanneer er wijzigingen in de code worden aangebracht.

Start het programma om het volgende te starten REST webservice:

```
(venv) [apinelli@centos8_cxlabs_spo app]$ python connectionless.py
 * Serving Flask app 'connectionless' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://10.122.28.2:8080/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 117-025-064
```

Abonnement op een client zonder verbinding

De gebruiker abonneert zich op meldingen: RESTService-endpoint wordt samen met het onderwerp verzonden om te abonneren op. In dit geval is het onderwerp all.

```
curl --location -X POST --insecure 'https://10.122.28.3/restconf/data/v1/cisco-notifications:subscriptio
--header 'Accept: application/json' \
--header 'Content-Type: application-json' \
--header 'Authorization: Basic cm9vdDpQYXNzMTIzNA==' \
--data '{
    "push.endpoint-url":"http://10.122.28.2:8080/api/posts",
    "push.topic":"all",
    "push.format": "json"
}'
```

Het verwachte antwoord is een antwoord in 2010, samen met de details van het abonnement in de kern van het antwoord:

```
{
   "push.notification-subscription":{
      "push.subscription-id":6243853653106271664,
      "push.subscribed-user":"root",
      "push.endpoint-url":"http:\/\/10.122.28.2:8080\/api\/posts",
      "push.topic":"all",
      "push.creation-time":"Fri Mar 31 17:07:48 BRT 2023",
      "push.creation-time-iso8601":"2023-03-31T17:07:48.159-03:00",
      "push.time-of-update":"Fri Mar 31 17:07:48 BRT 2023",
      "push.time-of-update-iso8601":"2023-03-31T17:07:48.159-03:00",
      "push.format":"json",
      "push.connection-type":"connection-less"
   }
}
```


Het is mogelijk om de lijst met meldingen waarop de gebruiker is geabonneerd te krijgen met een GET-verzoek:

```
curl --location --insecure 'https://10.122.28.3/restconf/data/v1/cisco-notifications:subscription' \
--header 'Accept: application/json' \
--header 'Content-Type: application-json' \
--header 'Authorization: Basic cm9vdDpQYXNzMTIzNA=='
```

De respons is als volgt:


```
{
   "com.response-message":{
      "com.header":{
         "com.firstIndex":0,
         "com.lastIndex":0
      },
      "com.data":{
         "push.notification-subscription":{
            "push.subscription-id":6243853653106271664,
            "push.subscribed-user":"root",
            "push.endpoint-url":"http:\/\/10.122.28.2:8080\/api\/posts",
            "push.session-id":0,
            "push.topic":"all",
            "push.creation-time":"Fri Mar 31 17:07:48 BRT 2023",
            "push.time-of-update":"Fri Mar 31 17:07:48 BRT 2023",
            "push.format":"json",
            "push.connection-type":"connection-less"
         }
      }
   }
}
```


#2

Verificatie van berichten, DEBUG-vermeldingen; show log, **Bestandsnaam gebruikt, SQL Outputs**

Bericht uit de reactie dat er één abonnement is voor all **("push.topic": "all")**. U kunt het bevestigen met een query naar de database (merk op dat het type abonnement 'zonder verbinding' is en de SUBSCRIPTIONID komt overeen met de uitvoer van de GET opdracht (geel gemarkeerd):

Als u een abonnement zonder verbinding moet verwijderen, kunt u een HTTP VERWIJDEREN aanvraag, met de abonnement-id die u wilt verwijderen. Stel dat u **abonnement-id** 6243853653106271664 wilt verwijderen:

```
curl --location --insecure --request DELETE 'https://10.122.28.3/restconf/data/v1/cisco-notifications:su
--header 'Accept: application/json' \
--header 'Content-Type: application-json' \
--header 'Authorization: Basic cm9vdDpQYXNzMTIzNA=='
```

Nu als u de DB opnieuw vragen ziet u geen ingangen:

Wanneer een wijziging in de inventaris optreedt, drukt de klant de meldingen (die van hetzelfde type zijn als de connection-oriented meldingen in het gedeelte over connected-oriented klanten), gevolgd door het antwoord van 2010:

```
(venv) [apinelli@centos8_cxlabs_spo app]$ python connectionless.py
 * Serving Flask app 'connectionless' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://10.122.28.2:8080/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 117-025-064
{'push.push-change-update': {'push.notification-id': -2185938612268228828, 'push.topic': 'inventory', 'p
10.122.28.3 - - [31/Mar/2023 16:47:23] "POST /api/posts HTTP/1.1" 201 -
{'push.push-change-update': {'push.notification-id': -1634959052215805274, 'push.topic': 'inventory', 'p
10.122.28.3 - - [31/Mar/2023 16:47:27] "POST /api/posts HTTP/1.1" 201 -
```

Conclusie

De twee soorten API-gebaseerde meldingen die in EPNM kunnen worden geconfigureerd (connectionlessen connection-oriented) worden toegelicht en de voorbeelden van de respectieve cliënten die als basis voor simulatiedoeleinden kunnen worden gebruikt, worden gegeven.

Referenties

- https://<fqdn-epnm>/nbi_help/component.html?comp_id=Notification%20Abonnementen%20Retrieval&api=restconf
- https://www.cisco.com/c/dam/en/us/td/docs/net_mgmt/epn_manager/RESTConf/Cisco_EPN_Manager_RESTConf_NBI_Guide_5_1_2.zip
- https://www.cisco.com/c/dam/en/us/td/docs/net_mgmt/epn_manager/RESTConf/Cisco_Evolved_Programmable_Network_Manager_5_1_2_REST
- Technische ondersteuning en documentatie â€“ Cisco Systems

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.