API 기반 EPNM 알림 문제 해결

목차

소개

이 문서에서는 REST API를 사용하여 디바이스 결함 정보에 액세스하는 경우 EPNM 알림 문제를 해결하는 방법에 대해 설명합니다.

배경 정보

구현하는 클라이언트는 알림을 전송하기 위해 EPNM(Evolved Programmable Network Manager)에서 사용되는 두 가지 메커니즘 중 하나를 처리하고 가입할 수 있어야 합니다.

EPNM API 알림

알림은 네트워크 관리자와 운영자에게 네트워크와 관련된 중요한 이벤트 또는 문제에 대해 알립니다. 이러한 알림은 잠재적인 문제를 신속하게 탐지 및 해결하여 다운타임을 줄이고 전체적인 네트워크 성능을 향상시킵니다.

EPNM은 전자 메일을 통한 알림, 지정된 수신자에 대한 SNMP(Simple Network Management Protocol) 트랩 또는 외부 Syslog 서버에 대한 Syslog 메시지 등 다양한 방법을 처리할 수 있습니다. EPNM은 이러한 방법 외에도 인벤토리, 알람, 서비스 활성화, 템플릿 실행 및 고가용성에 대한 정보를 검색하기 위해 사용할 수 있는 REST API(Representational State Transfer Application Programming Interface)를 제공합니다.

API 기반 알림은 현재 두 가지 메커니즘을 사용하여 지원됩니다.

- 연결 지향 알림: 클라이언트가 미리 정의된 URL에 가입하고 보안 HTTPS 채널을 통한 기본

인증으로 WebSocket 클라이언트를 사용합니다.
- 연결 없는 알림: 사용자는 XML(Extensible Markup Language) 및/또는 JSON(JavaScript Object Notation) 페이로드를 POST 요청으로 수락할 수 있는 REST 웹 서비스를 사용해야 합니다.

모든 알림은 동일한 스키마를 공유하며 JSON 또는 XML 형식으로 검색할 수 있습니다.

기본 EPNM 컨피그레이션

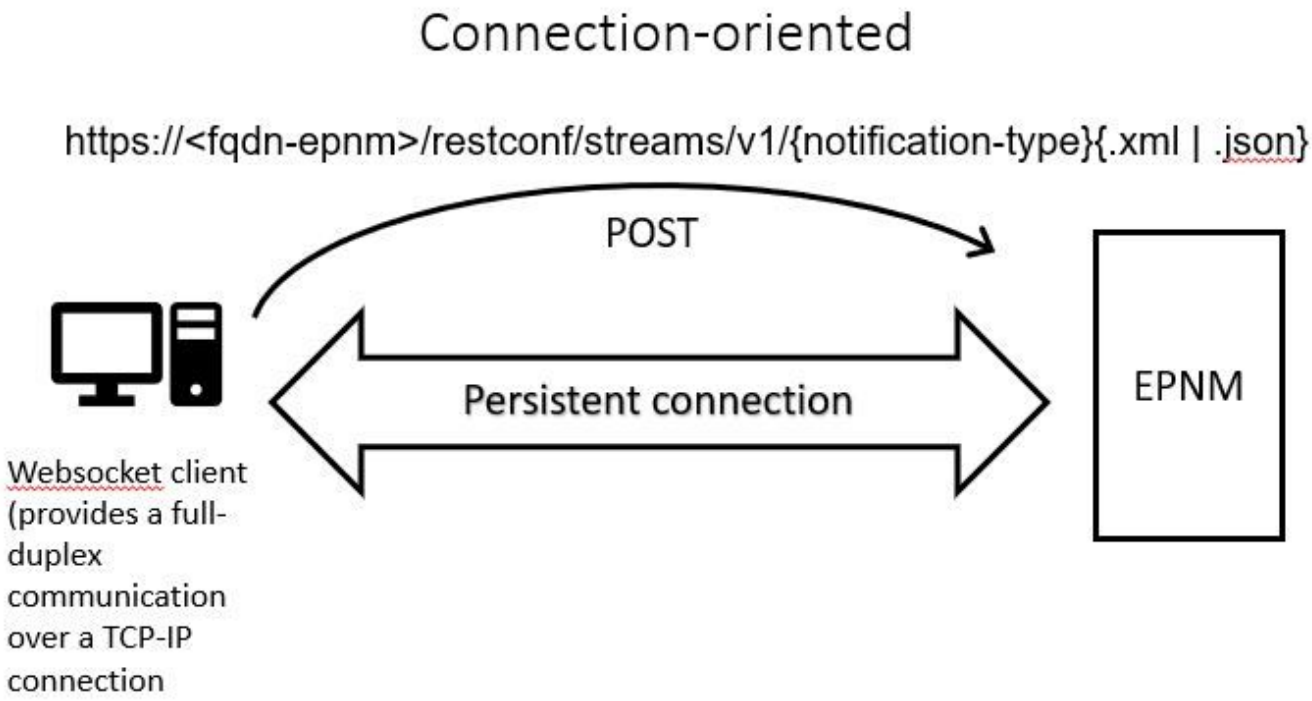
기본적으로 경보 및 인벤토리 알림은 비활성화되어 있습니다. 이를 활성화하려면 restconf-config.properties 표시된 대로 파일(EPNM 응용 프로그램을 다시 시작할 필요가 없음):

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

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

연결 지향 알림

그림에서 클라이언트 머신은 WebSocket을 실행하고 기본 인증과 보안 HTTPS 채널을 통해 미리 정의된 URL로 EPNM에 가입합니다.



WebSockets Python 클라이언트 실행

Python의 WebSocket-client 라이브러리를 사용하여 클라이언트 컴퓨터에 WebSocket을 만들 수 있

습니다.

```
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})
```

연결 지향 클라이언트 구독

이 코드는 EPNM에 가입하는 WebSocket 클라이언트를 설정합니다.
wss://10.122.28.3/restconf/streams/v1/inventory.json. Python을 사용하여 WebSocket연결을 설정하고 수신 및 발신 메시지를 처리하기 위한 라이브러리 서브스크립션은 다음과 같을 수도 있습니다(어떤 종류의 알림에 서브스크립션할 것인지 기준).

- /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}

이 on_message, on_error 및 on_close 함수는 WebSocket 연결이 메시지를 받거나 오류가 발생하거나 닫힐 때 호출되는 콜백 함수입니다. 이 **on_open** 함수는 WebSocket 연결이 설정되어 사용할 준비가 되었을 때 호출되는 콜백입니다.

이 username 및 password 변수는 원격 서버에 액세스하는 데 필요한 로그인 자격 증명으로 설정됩니다. 그런 다음 이 자격 증명은 base64 모듈을 추가하고 WebSocket 요청의 헤더에 추가합니다.

이 run_forever 연결을 시작하고, 연결을 무한정 열어 두고, 서버에서 오는 메시지를 듣기 위해 WebSocket 개체에서 메서드가 호출됩니다. 이 sslopt 매개변수는 연결에 대한 SSL/TLS 옵션을 구성하는 데 사용됩니다. 이 CERT_NONE 플래그는 인증 검증을 비활성화합니다.

WebSocket에서 알림을 받을 준비가 되도록 하려면 코드를 실행합니다.

```
(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: YYYYYYYYYYY
Sec-WebSocket-Version: 13
Connection: Upgrade
Authorization: Basic XXXXXXXXXXXX


-----------------------
--- 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!
```

다음 DB 쿼리로 서버에 대한 알림 등록을 확인할 수 있습니다.

```
ade # ./sql_execution.sh "SELECT * from RstcnfNtfctnsSbscrptnMngr WHERE CONNECTIONTYPE = 'connection-or
```

Cisco의 Advanced Threat Defense conn-oriented.txt 파일(DB 쿼리의 결과)은 다음과 같은 도구를 사용하여 HTML로 변환할 수 있습니다 aha (여기서는 Ubuntu 시스템에 그 용도가 설명되어 있음):

```
devasc@labvm:~/tmp$ sudo apt-get install aha
devasc@labvm:~/tmp$ cat conn-oriented.txt | aha > conn-oriented.html
```

그런 다음 conn-oriented.html 브라우저의 파일:

ID	INSTANCE_VERSION	CLASSNAME	CONNECTIONTYPE	ENDPOINTURL	SUBSCRIBEDUSER	SUBSCRIPTIONCREATIONTIME	SUBSCRIPTIONID	SUBSCRIPTIONTOPIC	SUBSCRIPTIONUPDATETIME
2361930571	0	cnfNtfctnSbscrptnMngr3	connection-oriented	de4d9082-c05c-439b-a7e7-65016623fee1	root	Mon Aug 28 16:13:04 BRT 2023	3648313822269611499	inventory	Mon Aug 28 16:13:04 BRT 2023

EPNM 온라인 설명서는 일단 설정되면 애플리케이션의 라이프사이클 내내 동일한 연결이 유지됩니다.

- 클라이언트에서 서버와의 연결이 끊길 때까지
- 유지 보수를 위해 또는 장애 조치 중에 서버가 중단될 때까지

특정 서브스크립션을 삭제해야 하는 경우 HTTP DELETE 에 요청 SUBSCRIPTIONID URL에 지정 https:// . 예를 들면 다음과 같습니다.

```
devasc@labvm:~/tmp$ curl --location --insecure --request DELETE 'https://10.122.28.3/restconf/data/v1/c
```

메시지 확인, 디버그 항목, show log, 사용된 파일 이름, SQL 출력

연결 지향 메커니즘을 사용하는 클라이언트가 알림을 제대로 받지 못하는 이유를 트러블슈팅하기 위해 지정된 DB 쿼리를 실행하고 구독이 있는지 여부를 확인할 수 있습니다. 없는 경우 클라이언트 소유자에게 서브스크립션을 발급하도록 요청합니다.

그 동안 DEBUG 레벨을 com.cisco.nms.nbi.epnm.restconf.notifications.handler.NotificationsHandlerAdapter 서브스크립션이 전송될 때마다 이를 탐지할 수 있습니다.

```
ade # sudo /opt/CSCOlumos/bin/setLogLevel.sh com.cisco.nms.nbi.epnm.restconf.notifications.handler.Noti
```

서브스크립션이 전송된 후 WebSocket 클라이언트의 IP 주소가 있는 항목이 localhost_access_log.txt:

```
ade # zgrep -h '"GET /restconf/streams/.* HTTP/1.1" 101' $(ls -1t /opt/CSCOlumos/logs/localhost_access_
```

마지막으로, DB를 다시 확인합니다(타임스탬프가 의 항목과 일치하는지 확인) localhost_access_log.txt).

H	CONNECTIONTYPE	ENDPOINTURL	ISENDPOINTREACHABLE	NOTIFICATIONFORMAT	SUBSCRIBEDUSER	SUBSCRIPTIONCREATIONTIME	SUBSCRIPTIONID	SUBSCRIPTIOI
	connection-oriented	852a674a-e3d0-4ecc-8ea0-787af30f1305	0	json	root	Mon Aug 28 22:17:06 BRT 2023	8743274415177640 88	inventory

다음 로그는 서브스크립션에 대한 POST 요청이 전송되는 시간을 보여줍니다.

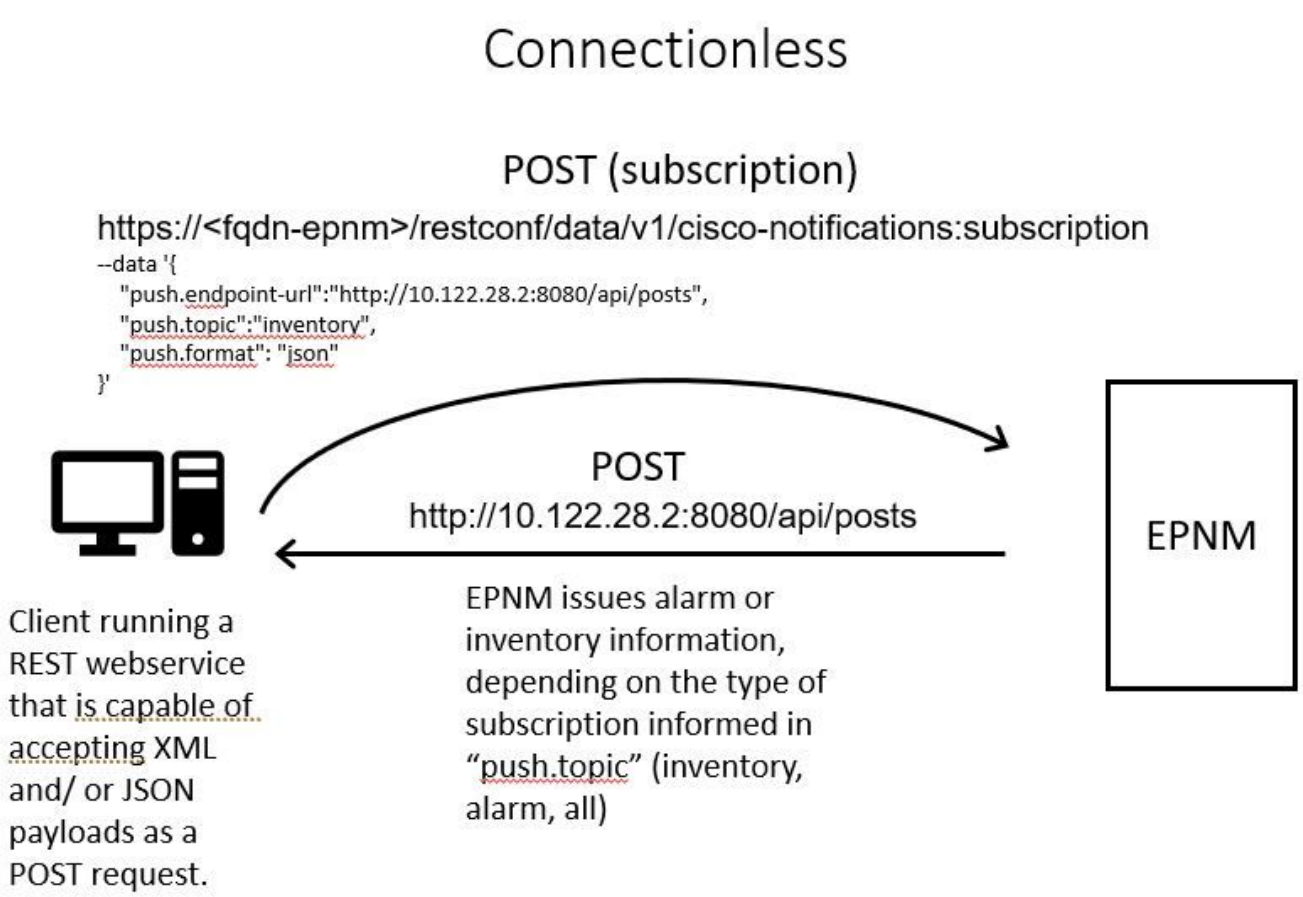
```
ade # grep -Eh 'DEBUG com.cisco.nms.nbi.epnm.restconf.notifications.handler.NotificationsHandlerAdapter
```

연결이 활성 상태로 유지되는 한 push-change-update 유형의 알림이 EPN-M 서버에서 알림을 구독하는 모든 클라이언트로 전송됩니다 . 이 예에서는 NCS2k의 호스트 이름이 변경될 때 EPNM에서 전송하는 알림 중 하나를 보여줍니다.

```
{ "push.push-change-update":{ "push.notification-id":2052931975556780123, "push.topic":"inventory", "pu
```

연결 없는 알림

다음은 의 경우 워크플로입니다. connectionless 알림을 전송합니다:



REST 웹 서비스 Python 클라이언트 실행

사용자는 XML 및/또는 JSON 페이로드를 POST 요청으로 수락할 수 있는 REST 웹 서비스를 사용해야 합니다. 이 REST 서비스는 Cisco EPNMrestconf 알림 프레임워크는 알림을 게시합니다. 이는 원격 시스템에 설치할 REST 웹 서비스의 예:

```
from flask import Flask, request, jsonify app = Flask(__name__) @ app.route('/api/posts', methods=['POS
```

단일 엔드포인트를 정의하는 Python Flask 웹 애플리케이션입니다 /api/posts ISE를 **HTTP POST** 요청입니다. 이 create_post() 함수가 호출될 때마다 **HTTP POST** 요청 대상 /api/posts. 내부 create_post() 함수, 들어오는 요청의 데이터는 **request.get_json()**- JSON 페이로드의 사전을 반환합니다. 그런 다음 페이로드에는 **print(post_data)** 디버그 목적으로. 그 후 키를 사용하여 응답 메시지가 생성됩니다 message 및 가치 **Post created successfully** (사전 형식). 이 응답 메시지는 HTTP 상태 코드 201(생성됨)을 사용하여 클라이언트에 반환됩니다.

이 if __name__ == '__main__': block은 스크립트가 모듈로 가져오지 않고 주 프로그램으로 실행되는지 여부를 확인하는 표준 Python 구성입니다. 스크립트가 주 프로그램으로 실행되면 Flask 애플리케이션을 시작하고 지정된 IP 주소 및 포트에서 실행됩니다. 이 **debug=True** 인수는 디버그 모드를 활성화합니다. 이 모드에서는 코드를 변경할 때 자세한 오류 메시지 및 서버의 자동 다시 로드를 제공합니다.

프로그램을 실행하여 REST 웹 서비스:

```
(venv) [apinelli@centos8_cxlabs_spo app]$ python connectionless.py * Serving Flask app 'connectionless'
```

연결 없는 클라이언트 서브스크립션

사용자가 알림에 가입합니다. REST서비스 엔드포인트는 서브스크립션을 위해 항목과 함께 전송됩니다. 이 경우 주제는 all.

```
[apinelli@centos8_cxlabs_spo ~]$ curl --location -X POST --insecure 'https://10.122.28.3/restconf/data/
```

예상 응답은 201 응답이며, 응답 본문의 서브스크립션 세부 정보도 함께 제공됩니다.

```
{ "push.notification-subscription": { "push.subscription-id": 7969974728822328535, "push.subscribed-use
```

GET 요청으로 사용자가 가입한 알림 목록을 가져올 수 있습니다.

```
curl --location --insecure 'https://10.122.28.3/restconf/data/v1/cisco-notifications:subscription' \ --
```

다음과 같은 응답을 산출했습니다.

```
{ "com.response-message": { "com.header": { "com.firstIndex": 0, "com.lastIndex": 1 }, "com.data": { "p
```

메시지 확인, 디버그 항목, show log, 사용된 파일 이름, SQL 출력

응답에서 두 개의 서브스크립션이 있음을 확인합니다. all (**"push.topic": "all"**) 인벤토리 관리 (**"push.topic": "inventory"**). 데이터베이스에 대한 쿼리로 이를 확인할 수 있습니다(서브스크립션 유형이 '연결 없음'이고 SUBSCRIPTIONID 필드는 의 출력과 일치하며 GET 노란색으로 강조 표시된 명령):

```
ade # ./sql_execution.sh "SELECT * from RstcnfNtfctnsSbscrptnMngr WHERE CONNECTIONTYPE = 'connection-le
```

```
        ID  INSTANCE_VERSION CLASSNAME                CONNECTIONTYPE   ENDPOINTURL                       SUBSCRIBEDUSER  SUBSCRIPTIONCREATIONTIME          SUBSCRIPTIONID       SUBSCRIPTIONTOPIC
----------  ---------------- ------------------------ ---------------- --------------------------------- --------------  -------------------------------- -------------------- --------------------
2361930573                0 cnfNtfctnsSbscrptnMngr3  connection-less  http://10.122.28.2:8080/api/posts root            Tue Aug 29 10:02:05 BRT 2023 7969974728822328535  all
 337897630                0 cnfNtfctnsSbscrptnMngr3  connection-less  http://10.122.28.2:8080/api/posts root            Fri Mar 31 17:45:47 BRT 2023 2985507860170167151  inventory
```

연결 없는 구독을 삭제해야 하는 경우 HTTP DELETE 삭제할 서브스크립션 ID를 사용하여 요청합니다. 다음을 삭제한다고 가정합니다. **subscription-id 2985507860170167151**:

```
curl --location --insecure --request DELETE 'https://10.122.28.3/restconf/data/v1/cisco-notifications:s
```

이제 DB를 다시 쿼리하면 SUBSCRIPTIONID 같음 7969974728822328535.

인벤토리가 변경되면 클라이언트에서는 알림이 출력됩니다. 알림은 connection-oriented 알림에 대한 섹션에서 볼 수 있습니다. connected-oriented 클라이언트), 201 응답:

```
(venv) [apinelli@centos8_cxlabs_spo app]$ python connectionless.py * Serving Flask app 'connectionless'
```

결론

이 문서에서는 EPNM에서 구성할 수 있는 두 가지 유형의 API 기반 알림(connectionless및 connection-oriented)에 대해 설명하고 시뮬레이션을 위한 기반으로 사용할 수 있는 각 클라이언트의 예를 들어 보겠습니다.

관련 정보

- https://www.cisco.com/c/dam/en/us/td/docs/net_mgmt/epn_manager/RESTConf/Cisco_Evolved_Programmable_Network_Manager_5_1_2_R
- 기술 지원 및 문서 – Cisco Systems

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.