プライベートクラウド上のイベントストリームのトラブルシューティング

内容

概要

このドキュメントでは、Advanced Malware Protection(AMP)セキュアエンドポイントプライベートクラウドのイベントストリームをトラブルシューティングする方法について説明します。

前提条件

要件

次の項目に関する知識があることが推奨されます。

- セキュアエンドポイントプライベートクラウド
- APIクエリー

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- セキュアエンドポイントプライベートクラウドv3.9.0
- cURL v7.87.0
- cURL v8.0.1

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

コンフィギュレーション

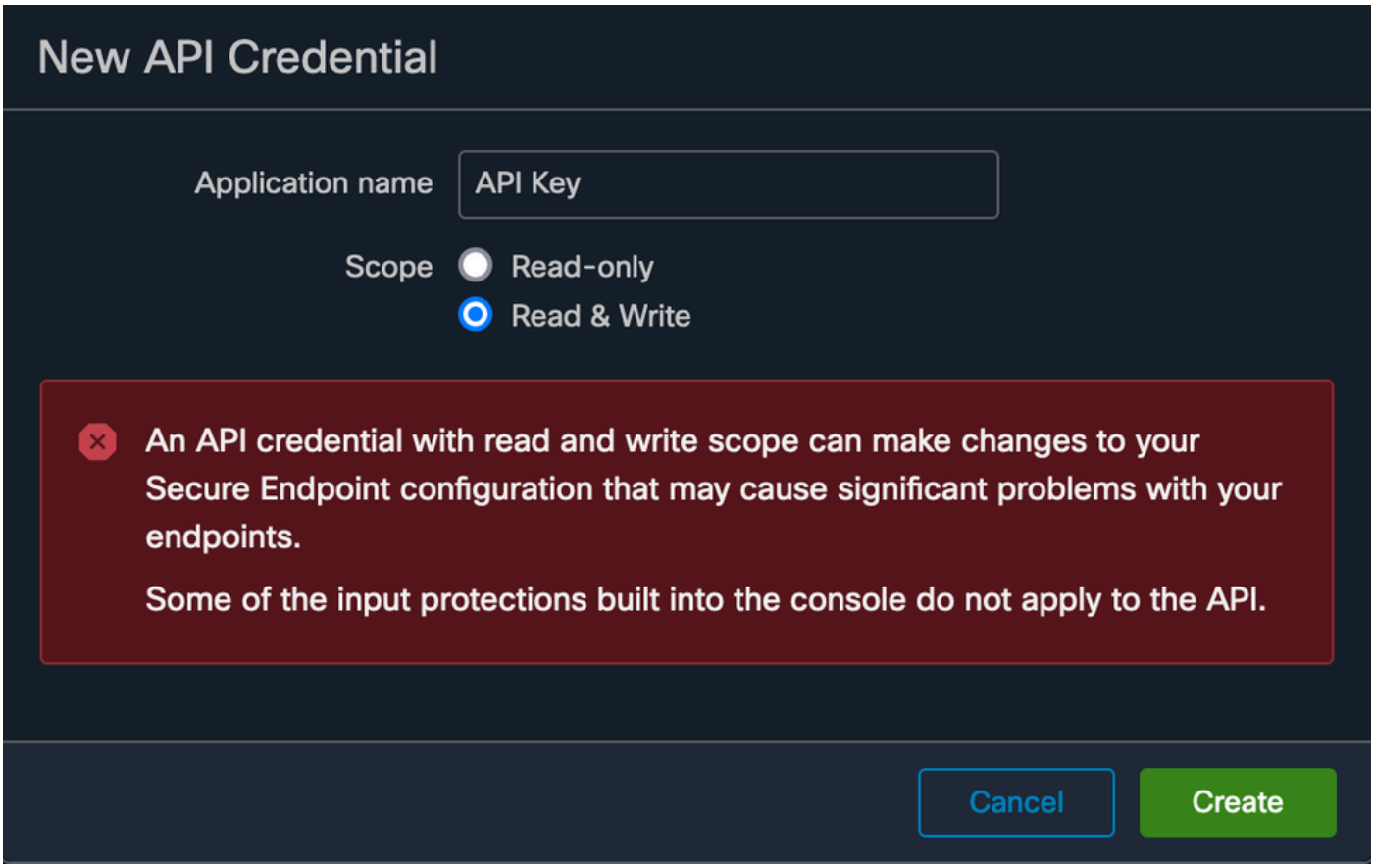
APIキーの作成

ステップ 1：プライベートクラウドコンソールにログインします。

ステップ 2：移動先 Accounts > API Credentialsを参照。

ステップ 3：クリック New API Credentialを参照。

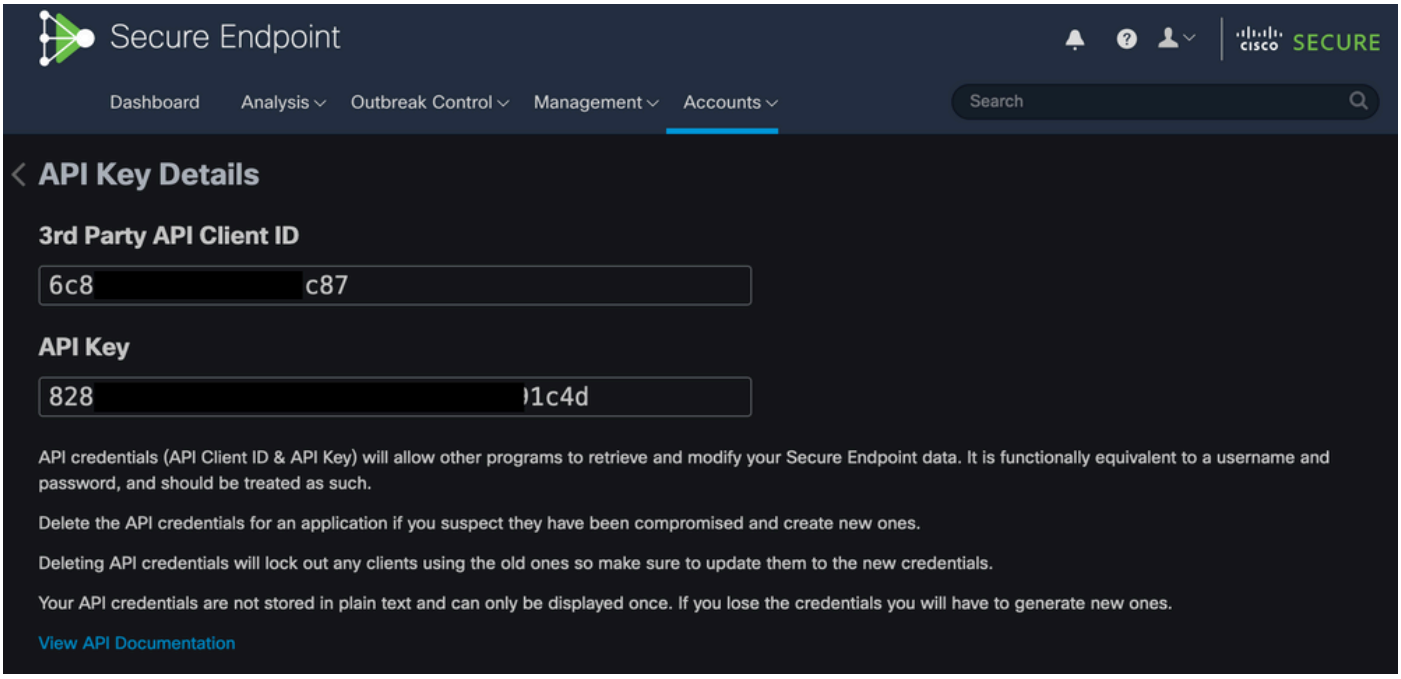
ステップ 4：次を追加します。 Application name をクリックして Read & Write 対象範囲.



APIキーの作成

ステップ 5：クリック Createを参照。

手順 6：APIクレデンシャルを保存します。



APIキー

注意：このページを終了すると、APIキーを回復できません。

イベントストリームの作成

これにより、イベント情報用の新しいAdvanced Message Queuing Protocol(AMQP)メッセージストリームが作成されます。

指定したイベントタイプおよびグループのイベントストリームを作成できます。

```
--data '{"name":"EVENT_STREAM_NAME","event_type":["EVENT_TYPE_1", "EVENT_TYPE_2"],"group_guid":["GROUP_
```

次の方法で、すべてのイベントタイプとすべてのグループのイベントストリームを作成できます。

```
--data '{"name":"EVENT_STREAM_NAME","event_type":[],"group_guid":[]}'
```

MacOS/Linux

MacOS/Linuxでイベントストリームを作成するには、次のコマンドを使用します。

```
curl -X POST -k -H 'Accept: application/json' -H 'Content-Type: application/json' -u 'CLIENT_ID:API_KEY
```

Windows

次のコマンドを使用して、Windows上でイベントストリームを作成できます。

```
curl -X POST -k -H "Accept: application/json" -H "Content-Type: application/json" -u "CLIENT_ID:API_KEY
```

応答

```
HTTP/1.1 201 Created
(...)
"data": {
  "id": 17,
  "name": "EVENT_STREAM_NAME",
  "amqp_credentials": {
```

```
    "user_name": "17-1bfXXXXXXXXX",
    "queue_name": "event_stream_17",
    "password": "3961XXXXXXXXXXXXXXXXXXXX814a77",
    "host": "FMC_SERVICE_URL",
    "port": 443,
    "proto": "https"
  }
}
```

イベントストリームのリスト

プライベートクラウドで作成されたイベントストリームのリストが表示されます。

MacOS/Linux

MacOS/Linuxでは、次の方法でイベントストリームをリストできます。

```
curl -k -H 'Accept: application/json' -H 'Content-Type: application/json' -u 'CLIENT_ID:API_KEY' -i 'ht
```

Windows

次のコマンドを使用して、Windows上のイベントストリームを一覧表示できます。

```
curl -k -H "Accept:application/json" -H "Content-Type:application/json" -u "CLIENT_ID:API_KEY" -i "http
```

応答

```
HTTP/1.1 200 OK
(...)
"data": {
  "id": 17,
  "name": "EVENT_STREAM_NAME",
  "amqp_credentials": {
    "user_name": "17-1bfXXXXXXXXX",
    "queue_name": "event_stream_17",
    "host": "FMC_SERVICE_URL",
    "port": 443,
    "proto": "https"
  }
}
```

イベントストリームの削除

アクティブなイベントストリームを削除します。

MacOS/Linux

MacOS/Linuxでは、次のコマンドを使用してイベントストリームを削除できます。

```
curl -X DELETE -k -H 'Accept: application/json' -H 'Content-Type: application/json' -u 'CLIENT_ID:API_K
```

Windows

次のコマンドを使用して、Windows上のイベントストリームを削除できます。

```
curl -X DELETE -k -H "Accept:application/json" -H "Content-Type:application/json" -u "CLIENT_ID:API_KEY
```

応答

```
HTTP/1.1 200 OK
(...)
 "data": {}
```

確認

ステップ 1：Pythonスクリプトをデバイスにコピーし、名前を付けて保存します EventStream.pyを参照。

```
import pika
import ssl

user_name = "USERNAME"
queue_name = "QUEUE_NAME"
password = "PASSWORD"
host = "FMC_SERVICE_URL"
port = 443
proto = "https"

def callback(channel, method, properties, body):
print(body)
```

```
amqp_url = f"amqps://{user_name}:{password}@{host}:{port}"

context = ssl.SSLContext(ssl.PROTOCOL_TLSv1_2)
amqp_ssl = pika.SSLOptions(context)

params = pika.URLParameters(amqp_url)
params.ssl_options = amqp_ssl

connection = pika.BlockingConnection(params)
channel = connection.channel()

channel.basic_consume(
   queue_name,
   callback,
   auto_ack = False
)

channel.start_consuming()
```

ステップ 2：端末で次のように実行します。 python3 EventStream.pyを参照。

ステップ 3：イベントストリームキューに追加されたイベントをトリガーします。

ステップ 4：イベントがターミナルに表示されるかどうかを確認します。

（「トラブルシューティング」）

これらのコマンドを実行するには、SSH経由でプライベートクラウドにログインする必要があります。

AMQPサービスの確認

サービスが有効になっているかどうかを確認します。

```
[root@fireamp rabbitmq]# amp-ctl service status rabbitmq
running enabled rabbitmq
```

サービスが実行されているかどうかを確認します。

```
[root@fireamp ~]# svstat /service/rabbitmq
/service/rabbitmq: up (pid 25504) 7402137 seconds
```

イベントストリームレシーバへの接続の確認

次のコマンドを実行します。

```
tail /data/log/rabbitmq/rabbit@fireamp.log
```

接続が確立されます。

```
=INFO REPORT==== 19-Apr-2023::08:40:12 ===
accepting AMQP connection <0.17588.27> (127.0.0.1:32946 -> 127.0.0.1:5672)
```

接続が閉じています：

```
=WARNING REPORT==== 19-Apr-2023::08:41:52 ===
closing AMQP connection <0.17588.27> (127.0.0.1:32946 -> 127.0.0.1:5672):
connection_closed_abruptly
```

キュー内のイベントの確認

キュー内のイベントは、接続が確立された後、このイベントストリームで受信者に送信される準備が整います。この例では、イベントストリームID 23に対して14のイベントがあります。

<#root>

```
[root@fireamp rabbitmq]# rabbitmqctl list_queues
Listing queues ...
1acb0eb6-39f7-4b11-bd9b-fc4dd0e3bd77_60b15rn8mpftaico6or6l8zxavl1usm 26
1acb0eb6-39f7-4b11-bd9b-fc4dd0e3bd77_61984nlu8p11eeopmgmtcjra1v8gf5p 26
1acb0eb6-39f7-4b11-bd9b-fc4dd0e3bd77_iesRAgVoOh287mO_DetOx9PdDu8MxkS6kL4oSTeBm9s 26
event_decoration 0
event_log_store 0
event_stream_23 14
event_streams_api 0
events_delayed 0
events_retry 0
mongo_event_consumer 0
out_events_q1 0
tevent_listener 0
```

ネットワークトラフィックファイルの収集

プライベートクラウドからのイベントストリームトラフィックを確認するには、 tcpdump ツール:

ステップ 1：プライベートクラウドにSSH接続します。

ステップ 2：次のコマンドを実行します。

```
tcpdump -vvv -i eth1 host <Event_Stream_Receiver_IP> -w file.pcap
```

ステップ 3：次のコマンドでキャプチャを停止します Ctrl+C (Windows)または Command-C (Mac)を使用します。

ステップ 4：を抽出します。 pcap プライベートクラウドからファイルを取得します

関連情報

- エンドポイント用AMPのイベントストリーム機能の設定
- テクニカル サポートとドキュメント - Cisco Systems

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。