# Dépannage du flux d'événements sur le cloud privé

## Table des matières



## Introduction

Ce document décrit comment dépanner les flux d'événements dans Advanced Malware Protection Secure Endpoint Private Cloud.

## Conditions préalables

### Exigences

Cisco vous recommande d'avoir des connaissances sur les sujets suivants :

- Cloud privé de terminal sécurisé
- requête API

### Composants utilisés

Les informations contenues dans ce document sont basées sur les versions de matériel et de logiciel suivantes :

- Cloud privé de terminal sécurisé v3.9.0
- cURL v7.87.0
- cURL v8.0.1

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l’incidence possible des commandes.

## Configuration

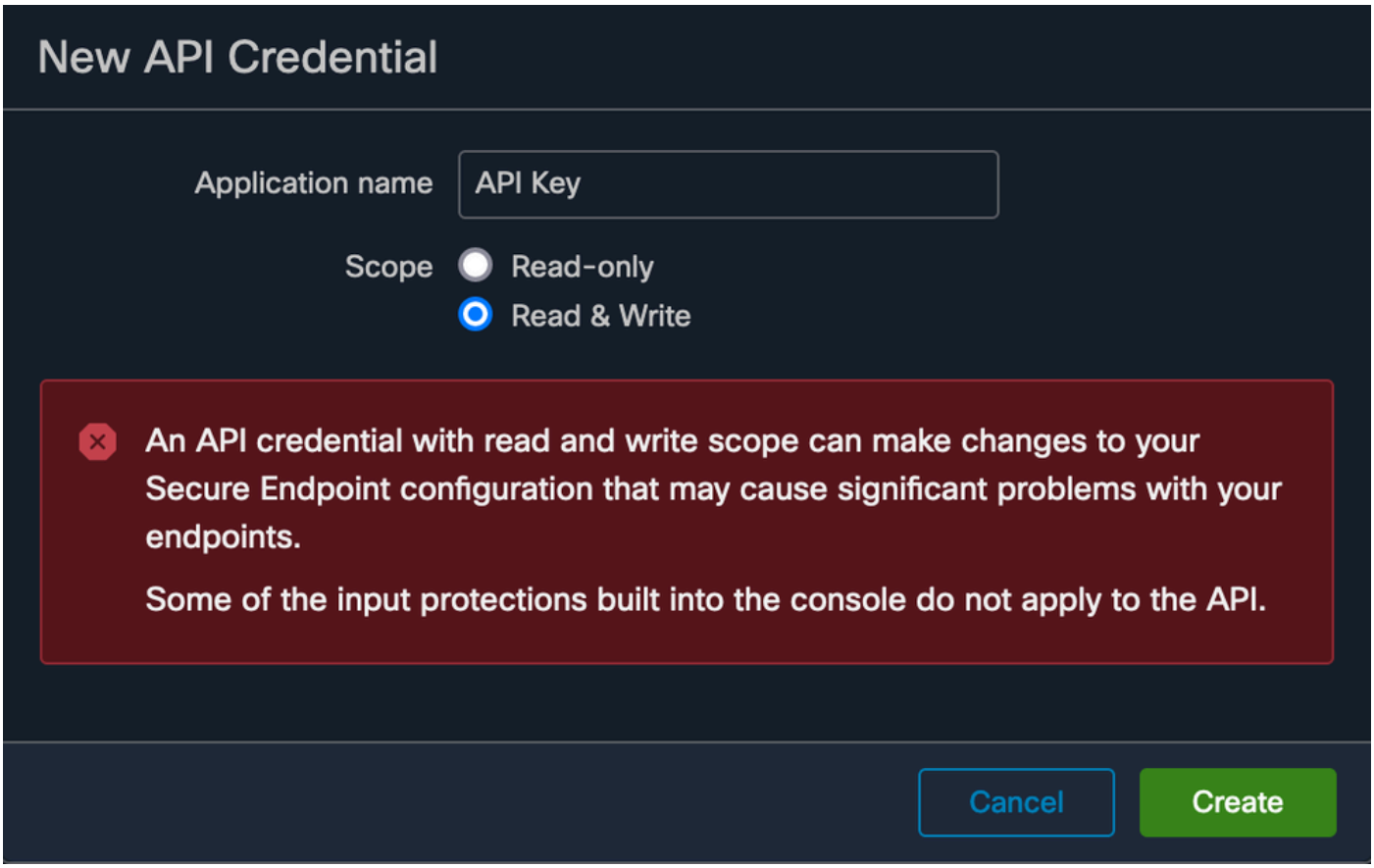
### Créer une clé API

Étape 1. Connectez-vous à la console de cloud privé.

Étape 2. Naviguez jusqu'à Accounts > API Credentials.

Étape 3. Cliquer New API Credential.

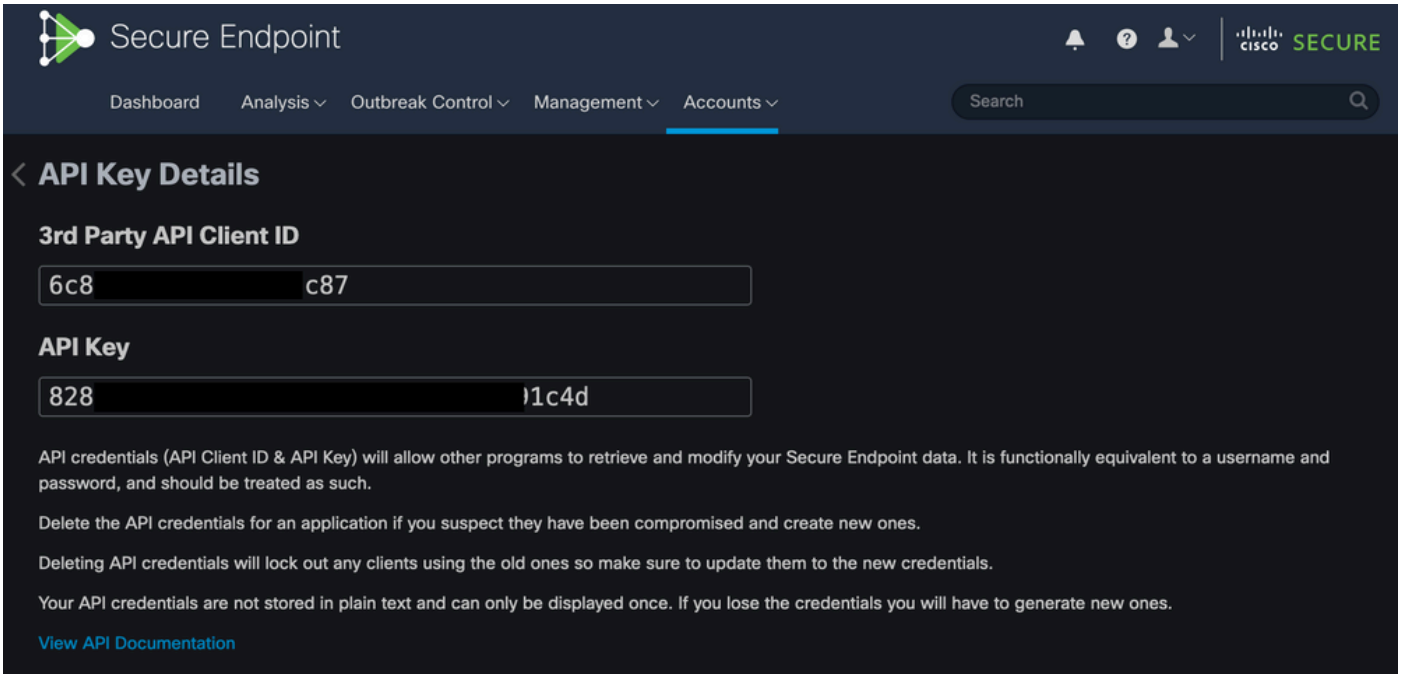
Étape 4. Ajoutez le Application name et cliquez sur Read & Write étendue.




Créer une clé API

Étape 5. Cliquer Create.

Étape 6. Enregistrez les identifiants API.




Clé API

Attention : la clé API ne peut pas être récupérée si vous quittez cette page.

## Créer un flux d'événements

Ceci crée un nouveau flux de messages AMQP (Advanced Message Queuing Protocol) pour les informations d'événement.

Vous pouvez créer un flux d'événements pour des types et des groupes d'événements spécifiés :

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

Vous pouvez créer un flux d'événements pour tous les types d'événements et tous les groupes en :

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

### MacOS/Linux

Vous pouvez créer un flux d'événements sur MacOS/Linux en utilisant :

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

### Fenêtres

Vous pouvez créer un flux d'événements sous Windows à l'aide de :

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

### Réponse

```
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"
  }
}
```

## Liste des flux d'événements

La liste des flux d'événements créés sur le cloud privé s'affiche.

### MacOS/Linux

Vous pouvez répertorier les flux d'événements sur MacOS/Linux en utilisant :

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

### Fenêtres

Vous pouvez répertorier les flux d'événements sous Windows à l'aide de :

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

### Réponse

```
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"
  }
}
```

### Supprimer les flux d'événements

Supprime un flux d'événements actif.

MacOS/Linux

Vous pouvez supprimer des flux d'événements sur MacOS/Linux à l'aide de :

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

Fenêtres

Vous pouvez supprimer des flux d'événements sous Windows à l'aide de :

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

Réponse

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

## Vérifier

Étape 1. Copiez le script Python sur votre périphérique et enregistrez-le sous EventStream.py.

```
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()
```

Étape 2. Exécutez-le dans le terminal en tant que `python3 EventStream.py`.

Étape 3. Déclenchez tout événement ajouté à la file d'attente Event Stream.

Étape 4. Vérifiez si les événements apparaissent dans le terminal.

# Dépannage

Pour exécuter ces commandes, vous devez vous connecter via SSH au cloud privé.

## Vérifier le service AMQP

Vérifiez si le service est activé :

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

Vérifiez si le service est en cours d'exécution :

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

## Vérifier la connexion au récepteur de flux d'événements

Exécutez la commande suivante :

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

La connexion est établie :

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

La connexion est fermée :

```
=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
```

## Rechercher les événements dans la file d'attente

Les événements de la file d'attente sont prêts à être envoyés sur ce flux d'événements au destinataire une fois la connexion établie. Dans cet exemple, il y a 14 événements pour Event Stream ID 23.

<#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
```

## Collecter le fichier de trafic réseau

Afin de vérifier le trafic Event Stream à partir du cloud privé, vous pouvez collecter des captures avec un tcpdump outil :

Étape 1. SSH dans le cloud privé.

Étape 2. Exécutez la commande suivante :

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

Étape 3. Arrêter la capture avec Ctrl+C (Windows) ou Command-C (Mac).

Étape 4. Extrayez le pcap à partir du cloud privé.

## Informations connexes

- Configuration de la fonctionnalité de flux d'événements AMP for Endpoints
- Assistance et documentation techniques - Cisco Systems

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.