# Automatiser le démarrage/l'arrêt de l'isolation sur plusieurs terminaux

## Table des matières


Introduction

Conditions préalables

- Exigences
- Composants utilisés

Informations générales

Problème

Solution

- Script
- Instructions

Vérifier


## Introduction

Ce document décrit comment automatiser l'isolation stop/start sur plusieurs terminaux à l'aide de l'API pour Cisco Secure Endpoint.

## Conditions préalables

### Exigences

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- Terminaux sécurisés Cisco
- Cisco Secure Endpoint Console
- API Cisco Secure Endpoint
- Python

### Composants utilisés

Les informations contenues dans ce document sont basées sur les versions logicielles suivantes :

- Terminal sécurisé Cisco 8.4.0.30201
- Point de terminaison vers environnement python hôte
- Python 3.11.7

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.

## Informations générales

- Vous utilisez une demande PUT pour lancer l'isolation.
- Une requête DELETE est utilisée pour arrêter l'isolement.
- Consultez la documentation de l'API pour plus d'informations.

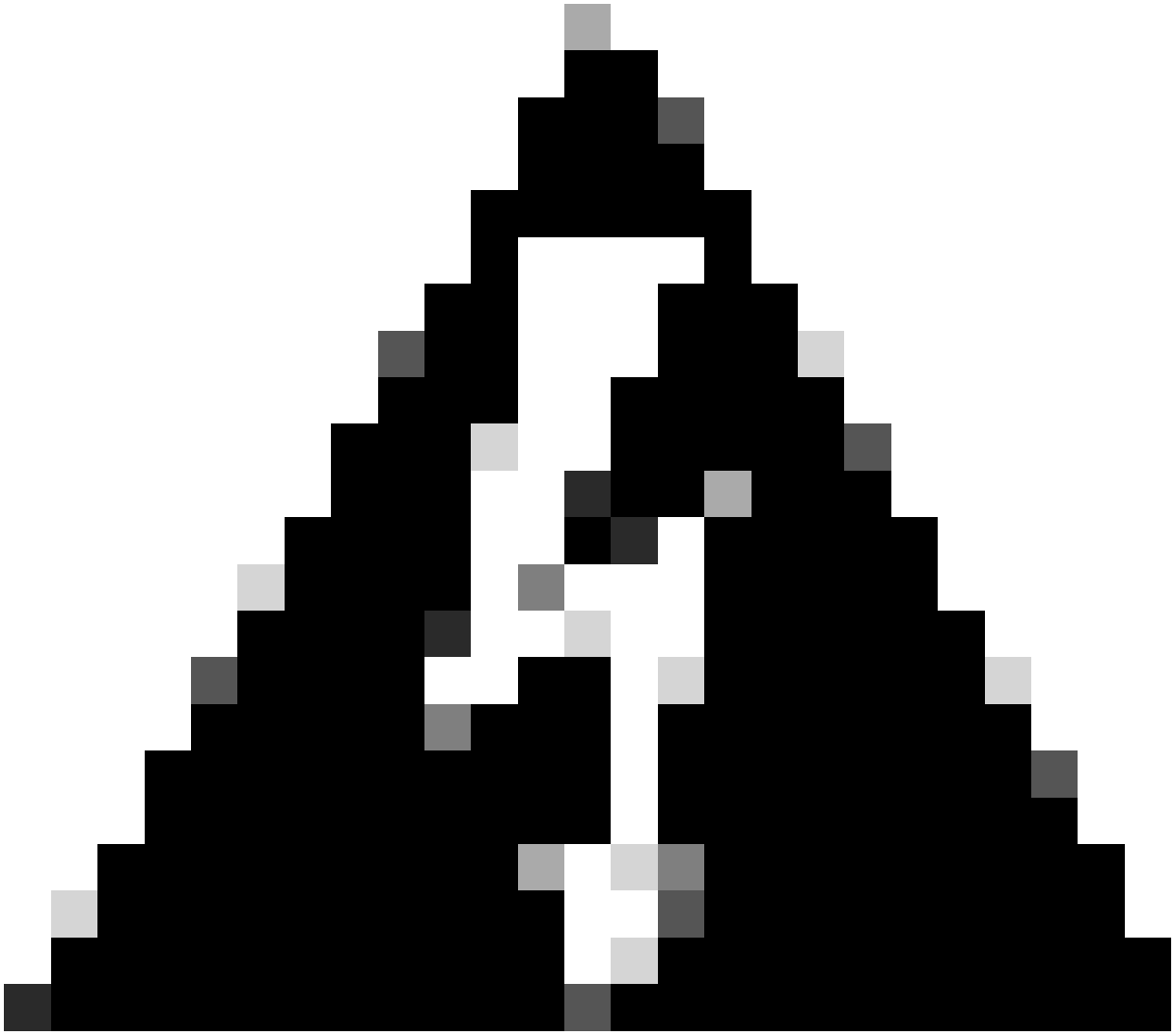
## Problème

Cisco Secure Endpoint permet de démarrer/arrêter l'isolation sur une machine à la fois. Cependant, lors d'un incident de sécurité, il est souvent nécessaire d'effectuer ces opérations sur plusieurs terminaux simultanément pour contenir efficacement les menaces potentielles. L'automatisation du processus d'isolation de début/fin pour les terminaux en masse à l'aide de l'API peut améliorer considérablement l'efficacité de la réponse aux incidents et réduire le risque global pour le réseau.

## Solution

- Le script Python fourni dans cet article peut être utilisé pour initier/mettre fin à l'isolement sur plusieurs terminaux de votre organisation à l'aide des informations d'identification de l'API Secure Endpoint.
- Pour générer les informations d'identification de l'API AMP, reportez-vous à Présentation de l'API Cisco AMP for Endpoints
- Pour utiliser le script fourni, vous devez installer python sur vos terminaux.
- Après l'installation de python, veuillez installer le module requêtes

```
pip install requests
```



Avertissement : le script est fourni uniquement à titre d'illustration et vise à démontrer comment automatiser la fonction d'isolation des points de terminaison à l'aide de l'API. Le Centre d'assistance technique Cisco (TAC) n'intervient pas dans le dépannage des problèmes liés à ce script. Les utilisateurs doivent faire preuve de prudence et tester le script en profondeur dans un environnement sécurisé avant de le déployer dans un environnement de production.

## Script

Vous pouvez utiliser le script fourni pour commencer l'isolation sur plusieurs terminaux de votre entreprise :

```
import requests

def read_config(file_path):
    """
    Reads the configuration file to get the API base URL, client ID, and API key.
```

```
    """
    config = {}
    try:
        with open(file_path, 'r') as file:
            for line in file:
                # Split each line into key and value based on '='
                key, value = line.strip().split('=')
                config[key] = value
    except FileNotFoundError:
        print(f"Error: Configuration file '{file_path}' not found.")
        exit(1)  # Exit the script if the file is not found
    except ValueError:
        print(f"Error: Configuration file '{file_path}' is incorrectly formatted.")
        exit(1)  # Exit the script if the file format is invalid
    return config

def read_guids(file_path):
    """
    Reads the file containing GUIDs for endpoints to be isolated.
    """
    try:
        with open(file_path, 'r') as file:
            # Read each line, strip whitespace, and ignore empty lines
            return [line.strip() for line in file if line.strip()]
    except FileNotFoundError:
        print(f"Error: GUIDs file '{file_path}' not found.")
        exit(1)  # Exit the script if the file is not found
    except Exception as e:
        print(f"Error: An unexpected error occurred while reading the GUIDs file: {e}")
        exit(1)  # Exit the script if an unexpected error occurs

def isolate_endpoint(base_url, client_id, api_key, connector_guid):
    """
    Sends a PUT request to isolate an endpoint identified by the connector GUID.
    Args:
        base_url (str): The base URL for the API.
        client_id (str): The API client ID for authentication.
        api_key (str): The API key for authentication.
        connector_guid (str): The GUID of the connector to be isolated.
    """
    url = f"{base_url}/{connector_guid}/isolation"
    try:
        # Send PUT request with authentication
        response = requests.put(url, auth=(client_id, api_key))
        response.raise_for_status()  # Raise an HTTPError for bad responses (4xx and 5xx)

        if response.status_code == 200:
            print(f"Successfully isolated endpoint: {connector_guid}")
        else:
            print(f"Failed to isolate endpoint: {connector_guid}. Status Code: {response.status_code}, R
    except requests.RequestException as e:
        print(f"Error: An error occurred while isolating the endpoint '{connector_guid}': {e}")

if __name__ == "__main__":
    # Read configuration values from the config file
    config = read_config('config.txt')

    # Read list of GUIDs from the GUIDs file
    connector_guids = read_guids('guids.txt')

    # Extract configuration values
    base_url = config.get('BASE_URL')
```

```
    api_client_id = config.get('API_CLIENT_ID')
    api_key = config.get('API_KEY')

    # Check if all required configuration values are present
    if not base_url or not api_client_id or not api_key:
        print("Error: Missing required configuration values.")
        exit(1)  # Exit the script if any configuration values are missing

    # Process each GUID by isolating the endpoint
    for guid in connector_guids:
        isolate_endpoint(base_url, api_client_id, api_key, guid)
```

## Instructions

- Pour générer les informations d'identification de l'API AMP, reportez-vous à Présentation de l'API Cisco AMP for Endpoints
- Utilisez l'URL BASE_URL mentionnée pour votre région :

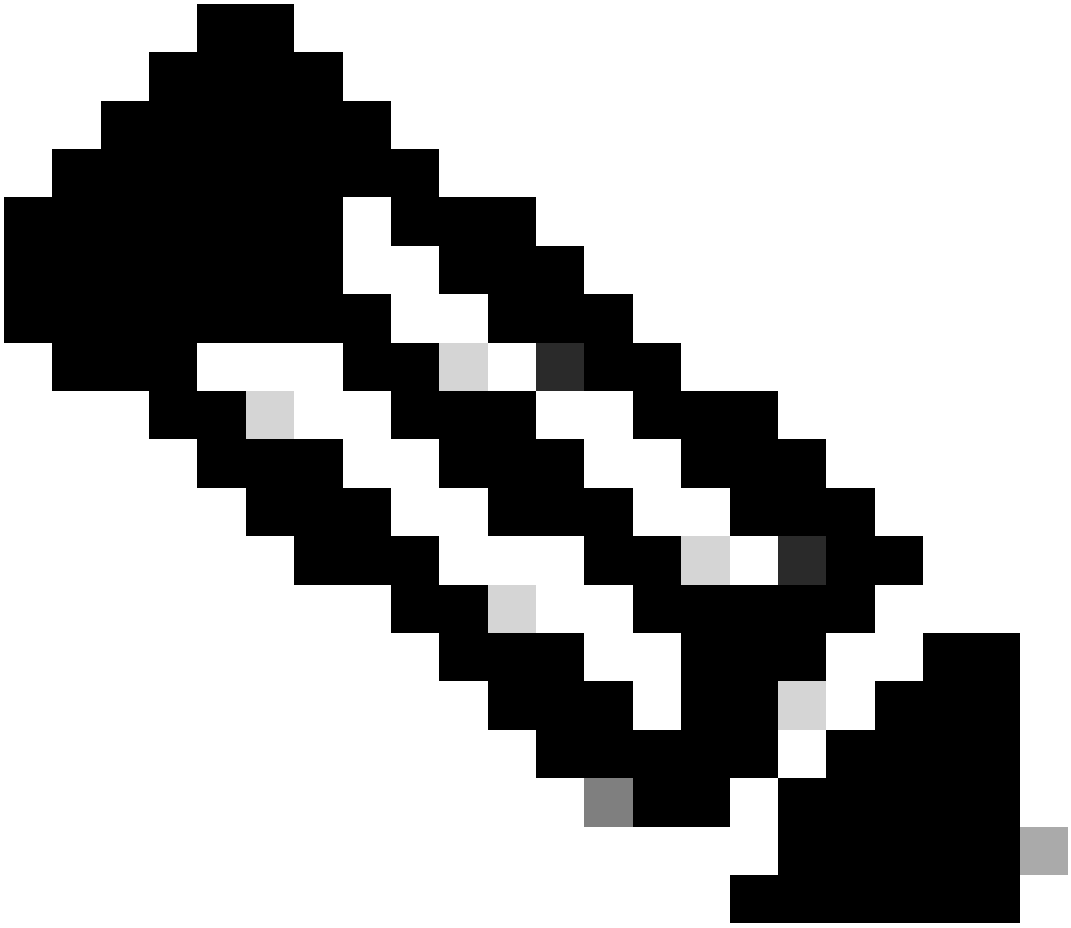
```
NAM - https://api.amp.cisco.com/v1/computers/
EU - https://api.eu.amp.cisco.com/v1/computers/
APJC - https://api.apjc.amp.cisco.com/v1/computers/
```

- Créez un fichier config.txt dans le même répertoire que le script avec le contenu mentionné. Exemple de fichier config.txt :

```
BASE_URL=https://api.apjc.amp.cisco.com/v1/computers/
API_CLIENT_ID=xxxxxxxxxxxxxxxxxxxx
API_KEY=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

- Créez un fichier guids.txt dans le même répertoire que le script avec la liste des GUID de connecteur, un par ligne. Ajoutez autant de GUID que nécessaire. Exemple de fichier guids.txt :

```
abXXXXXXXXXXXXcd-XefX-XghX-X12X-XXXXXX567XXXXXXX
yzXXXXXXXXXXXXlm-XprX-XmnX-X34X-XXXXXX618XXXXXXX
```



Remarque : vous pouvez collecter les GUID de vos terminaux via l'API GET /v1/computers ou depuis la console Cisco Secure Endpoint en accédant à Management > Computers, en développant l'entrée pour un terminal spécifique et en copiant le GUID du connecteur.

- Ouvrez un terminal ou une invite de commandes. Accédez au répertoire où start_isolation_script.py se trouve.
- Exécutez le script en exécutant la commande mentionnée :

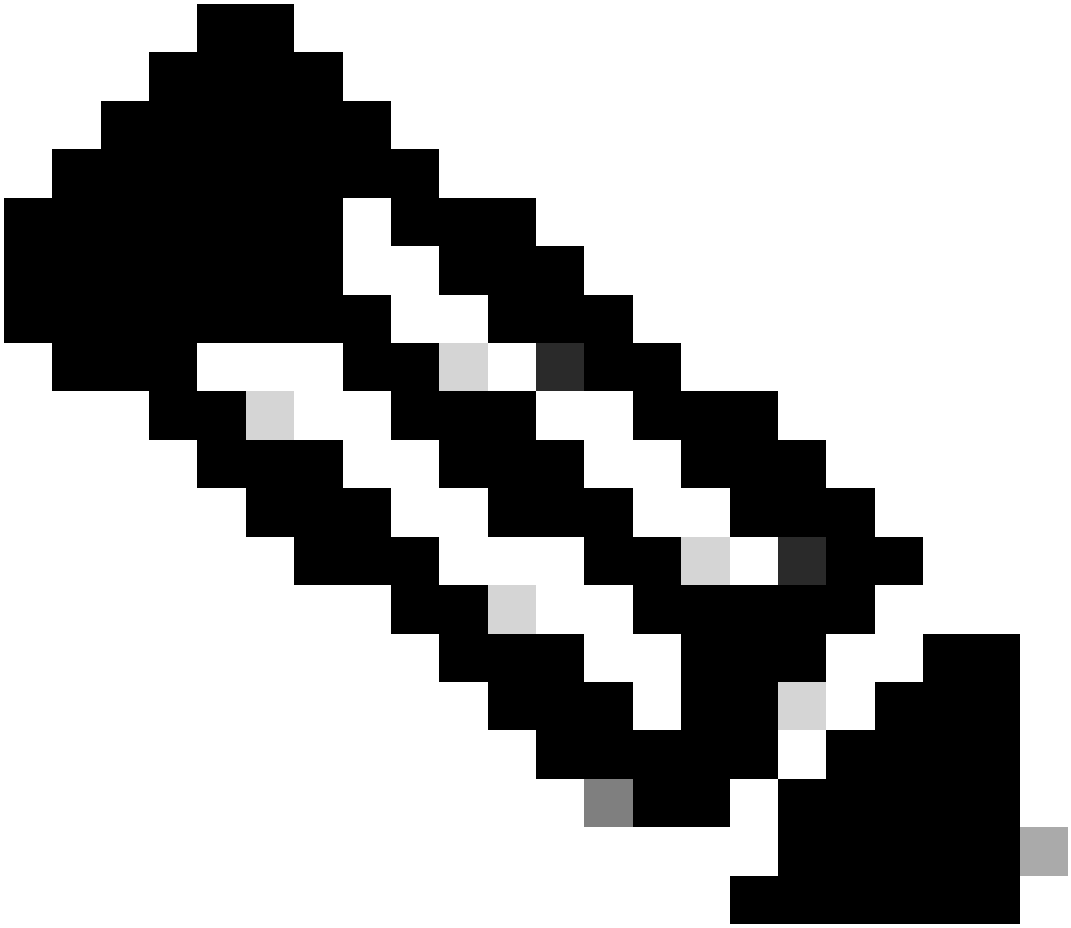
```
python start_isolation_script.py
```

## Vérifier

- Le script tente d'isoler chaque point de terminaison spécifié dans le fichier guids.txt.

- Vérifiez les messages de réussite ou d'erreur de chaque point d'extrémité dans le terminal ou l'invite de commande.

---



Remarque : le script joint start_isolation.py peut être utilisé pour lancer l'isolation sur les points d'extrémité, tandis que stop_isolation.py est conçu pour arrêter l'isolation sur les points d'extrémité. Toutes les instructions d'exécution et d'exécution du script restent identiques.

---

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.