在多個終端上自動啟動/停止隔離

目錄

簡介

本文檔介紹如何使用Cisco Secure Endpoint的API在多個終端上自動執行停止/啟動隔離。

必要條件

需求

思科建議您瞭解以下主題：

- 思科安全端點
- 思科安全終端主控台
- 思科安全終端API
- Python

採用元件

本檔案中的資訊是根據以下軟體版本：

- 思科安全終端8.4.0.30201
- 終端到託管python環境
- Python 3.11.7

本文中的資訊是根據特定實驗室環境內的裝置所建立。文中使用到的所有裝置皆從已清除（預設）的組態來啟動。如果您的網路運作中，請確保您瞭解任何指令可能造成的影響。

背景資訊

- 使用PUT請求啟動隔離。
- DELETE請求用於停止隔離。
- 有關詳細資訊，請參閱API文檔。

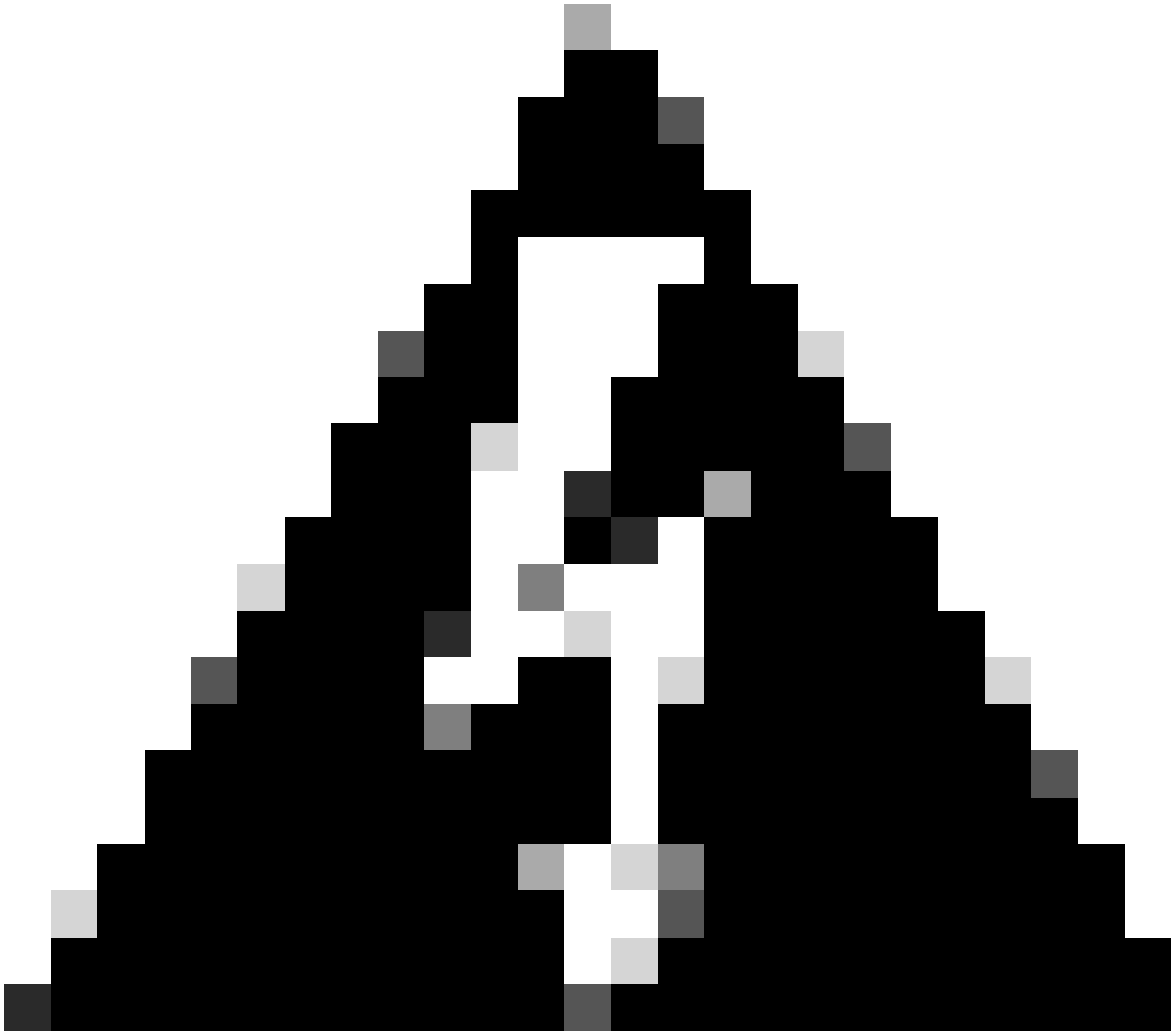
問題

思科安全終端允許一次在一台電腦上啟動/停止隔離。但是，在安全事件期間，通常需要在多個終端上同時執行這些操作，以有效遏制潛在威脅。使用API自動化批次端點的啟動/停止隔離過程可以顯著提高事件響應效率並降低對網路的總體風險。

解決方案

- 本文中提供的Python指令碼可用於使用安全終端API憑證在組織中的多個終端上啟動/終止隔離。
- 要生成AMP API憑證，請參閱面向終端的思科AMP的終端API概述
- 若要使用提供的指令碼，您需要在終端上安裝python。
- 安裝python之後，請安裝請求模組

```
pip install requests
```



警告：指令碼僅用於說明目的，旨在演示如何使用API自動執行終端隔離功能。思科技術協助中心(TAC)不參與此指令碼相關問題的疑難排解。使用者必須在安全環境中謹慎地全面測試該指令碼，然後再將其部署到生產環境中。

指令碼

您可以使用提供的指令碼在業務中的多個終端上開始隔離：

```
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}, 
    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)
```

說明

- 要生成AMP API憑證，請參閱面向終端的思科AMP的終端API概述
- 使用針對您所在區域提到的BASE_URL：

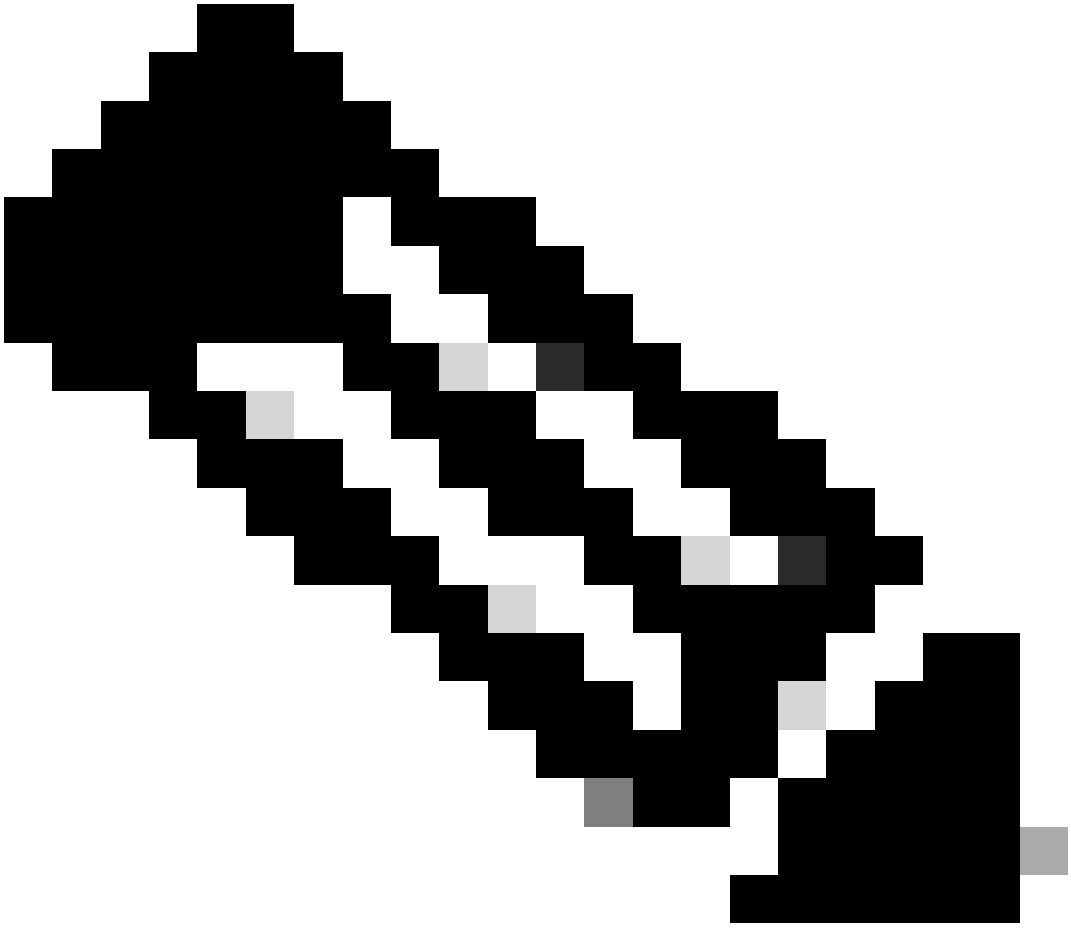
```
NAM - https://api.amp.cisco.com/v1/computers/
EU - https://api.eu.amp.cisco.com/v1/computers/
APJC - https://api.apjc.amp.cisco.com/v1/computers/
```

- 在與包含上述內容的指令碼相同的目錄中建立config.txt檔案。config.txt檔案範例：

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

- 在與指令碼相同的目錄中建立guids.txt檔案，其中包含聯結器GUID清單，每行一個。視需要新增GUID。guids.txt檔案範例：

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



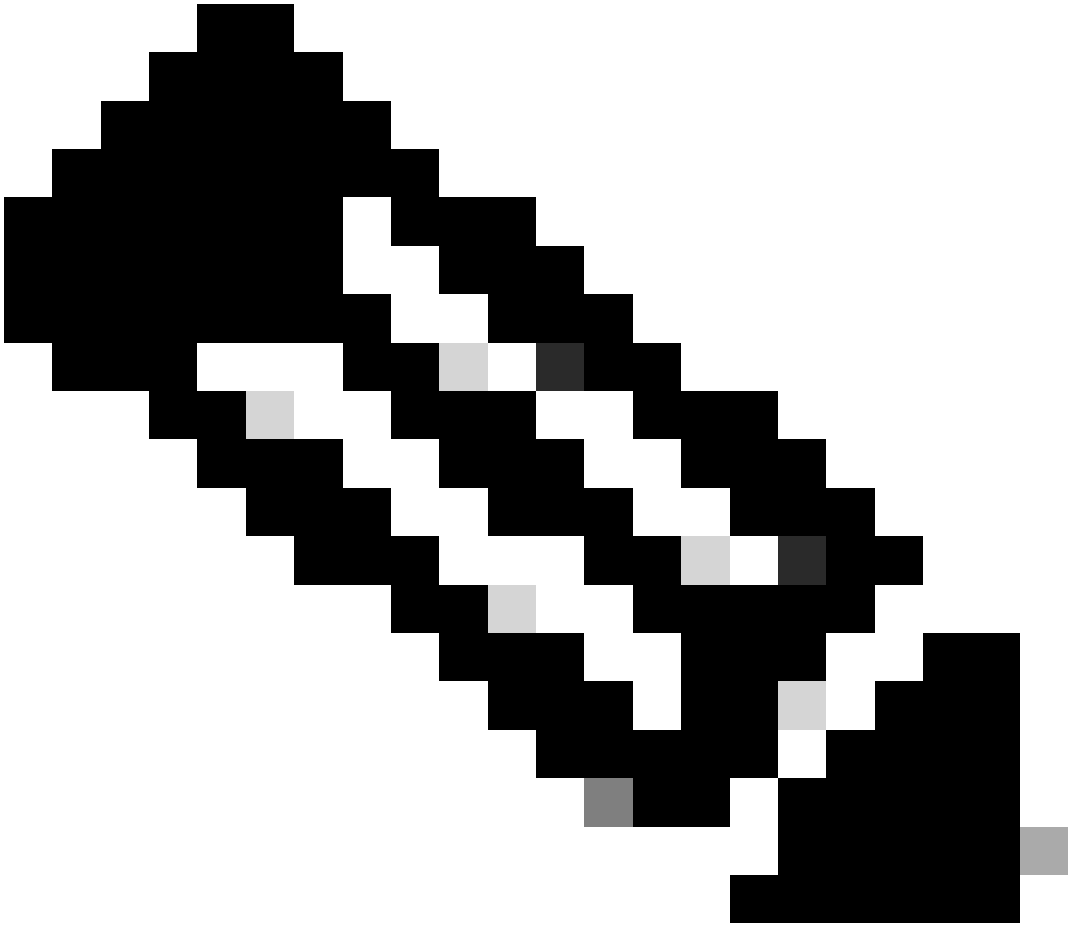
注意：您可以透過API GET /v1/computers或從Cisco Secure Endpoint Console收集終端的GUID，方法是導航到管理>電腦，展開特定終端的條目，然後複製聯結器GUID。

- 打開終端或命令提示符。導航到start_isolation_script.py所在的目錄。
- 透過運行所提及的命令執行指令碼：

```
python start_isolation_script.py
```

驗證

- 該指令碼嘗試隔離在guids.txt檔案中指定的每個端點。
- 檢查terminal或command prompt，瞭解每個端點的成功或錯誤消息。



注意：附加的指令碼start_isolation.py可用於在端點上啟動隔離，而stop_isolation.py旨在在端點上停止隔離。所有用於運行和執行指令碼的指令都保持不變。

關於此翻譯

思科已使用電腦和人工技術翻譯本文件，讓全世界的使用者能夠以自己的語言理解支援內容。請注意，即使是最佳機器翻譯，也不如專業譯者翻譯的內容準確。Cisco Systems, Inc. 對這些翻譯的準確度概不負責，並建議一律查看原始英文文件（提供連結）。