أتمتة بدء/إيقاف العزل على نقاط نهاية متعددة

المحتويات

المقدمة

يصف هذه المستند كيفية أتمتة عملية عزل الإيقاف/البدء على نقاط نهاية متعددة باستخدام واجهة برمجة التطبيقات (API) لنقطة النهاية الآمنة من Cisco.

المتطلبات الأساسية

المتطلبات

توصي Cisco بأن تكون لديك معرفة بالمواضيع التالية:

- نقطة نهاية آمنة من Cisco
- وحدة تحكم نقطة النهاية الآمنة من Cisco
- واجهة برمجة تطبيقات Cisco Secure Endpoint API
- بايثون

المكونات المستخدمة

تستند المعلومات الواردة في هذه المستند إلى إصدارات البرامج التالية:

- Cisco Secure Endpoint 8.4.0.30201
- نقطة النهاية لاستضافة بيئة بايثون
- Python 3.11.7

تم إنشاء المعلومات الواردة في هذه المستند من الأجهزة الموجودة في بيئة معملية خاصة. بدأت جميع الأجهزة المُستخدمة في هذه المستند بتكوين ممسوح (افتراضي). إذا كانت شبكتك قيد التشغيل، فتأكد من فهمك للتأثير المحتمل لأي أمر.

معلومات أساسية

- تستخدم طلب PUT لبدء العزل.
- يتم إستخدام طلب DELETE لإيقاف العزل.
- راجع وثائق واجهة برمجة التطبيقات للحصول على مزيد من المعلومات.

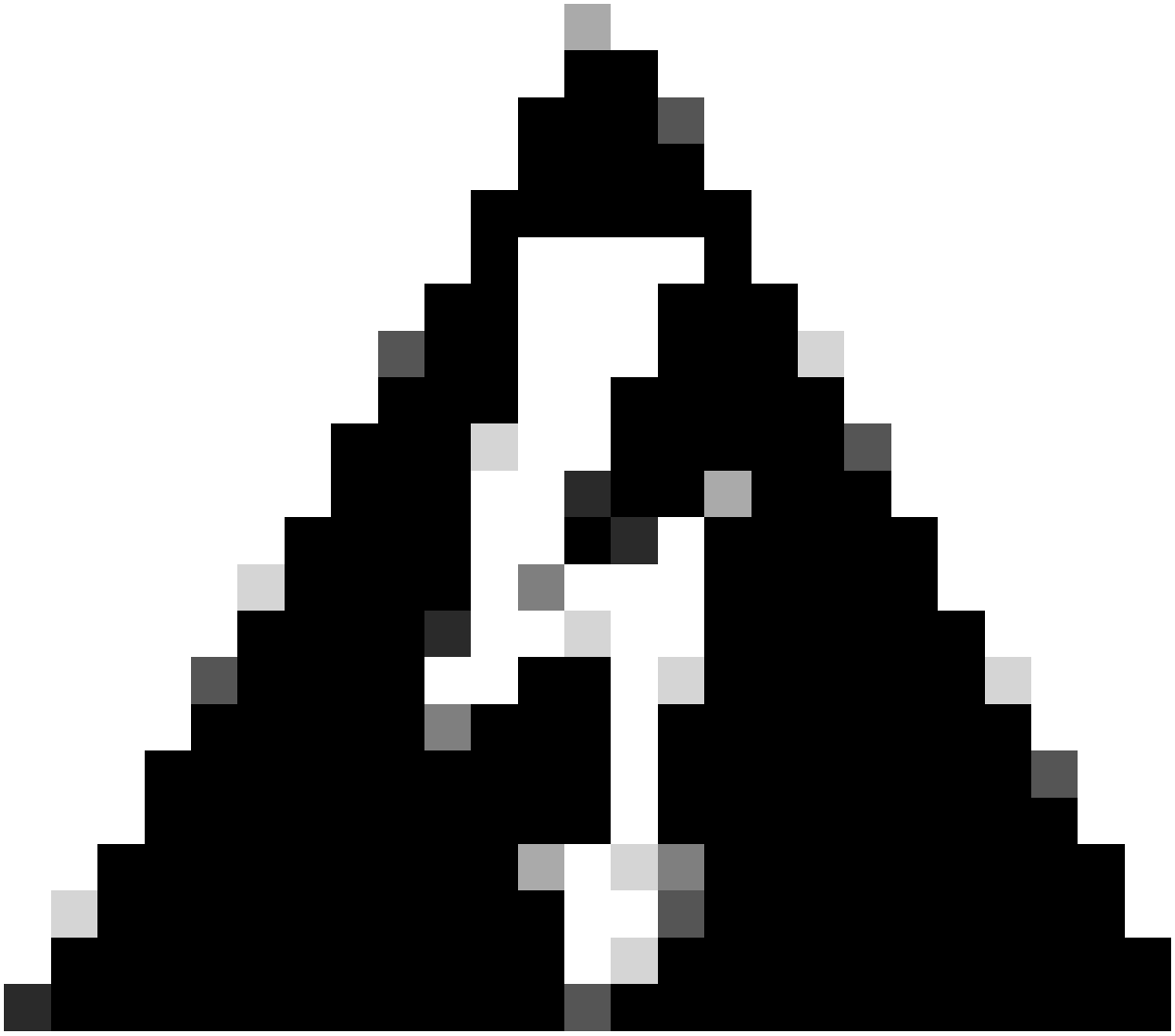
المشكلة

تتيح نقطة النهاية الآمنة من Cisco إمكانية عزل البدء/الإيقاف على جهاز واحد في كل مرة. ومع ذلك، خلال الحوادث الأمنية، عادة ما يكون من الضروري إجراء هذه العمليات على نقاط نهاية متعددة في نفس الوقت لاحتواء التهديدات المحتملة بشكل فعال. ويمكن أن تساعد أتمتة عملية العزل الخاصة ببدء/إيقاف التشغيل لنقاط النهاية الكبيرة باستخدام واجهة برمجة التطبيقات (API) على تحسين كفاءة الاستجابة للحوادث بشكل كبير، فضلا عن تقليل المخاطر الكلية التي تتعرض لها الشبكة.

الحل

- يمكن إستخدام برنامج Python النصي المتوفر في هذه المقالة لبدء/إنهاء العزل على نقاط نهاية متعددة في مؤسستك باستخدام بيانات اعتماد API لنقطة النهاية الآمنة.
- لإنشاء بيانات اعتماد AMP API، يرجى الرجوع إلى نظرة عامة على Cisco AMP للحصول على واجهة برمجة تطبيقات نقاط النهاية
- لاستخدام البرنامج النصي المتوفر، تحتاج إلى تثبيت Pythonon نقاط النهاية الخاصة بك.
- بعد تثبيت python، يرجى تثبيت الوحدة النمطية للطلبات

```
pip install requests
```



تحذير: يتم توفير البرنامج النصي لأغراض توضيحية فقط ويقصد به توضيح كيفية أتمتة ميزة عزل نقطة النهاية باستخدام واجهة برمجة التطبيقات (API). ال يشارك مركز المساعدة التقنية (TAC) من Cisco في أستكشاف أخطاء هذا البرنامج النصي وإصلاحها. يجب على المستخدمين توخي الحذر واختبار البرنامج النصي بدقة في بيئة آمنة قبل نشره في إعداد إنتاج.

نص

يمكنك إستخدام النص التنفيذي المتوفر لبدء العزل على نقاط نهاية متعددة في شركتك:

```
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)
```

تعليمات

- لإنشاء بيانات اعتماد AMP API، يرجى الرجوع إلى نظرة عامة على Cisco AMP للحصول على واجهة برمجة تطبيقات نقاط النهاية
- إستخدام BASE_URL المذكور في منطقتك:

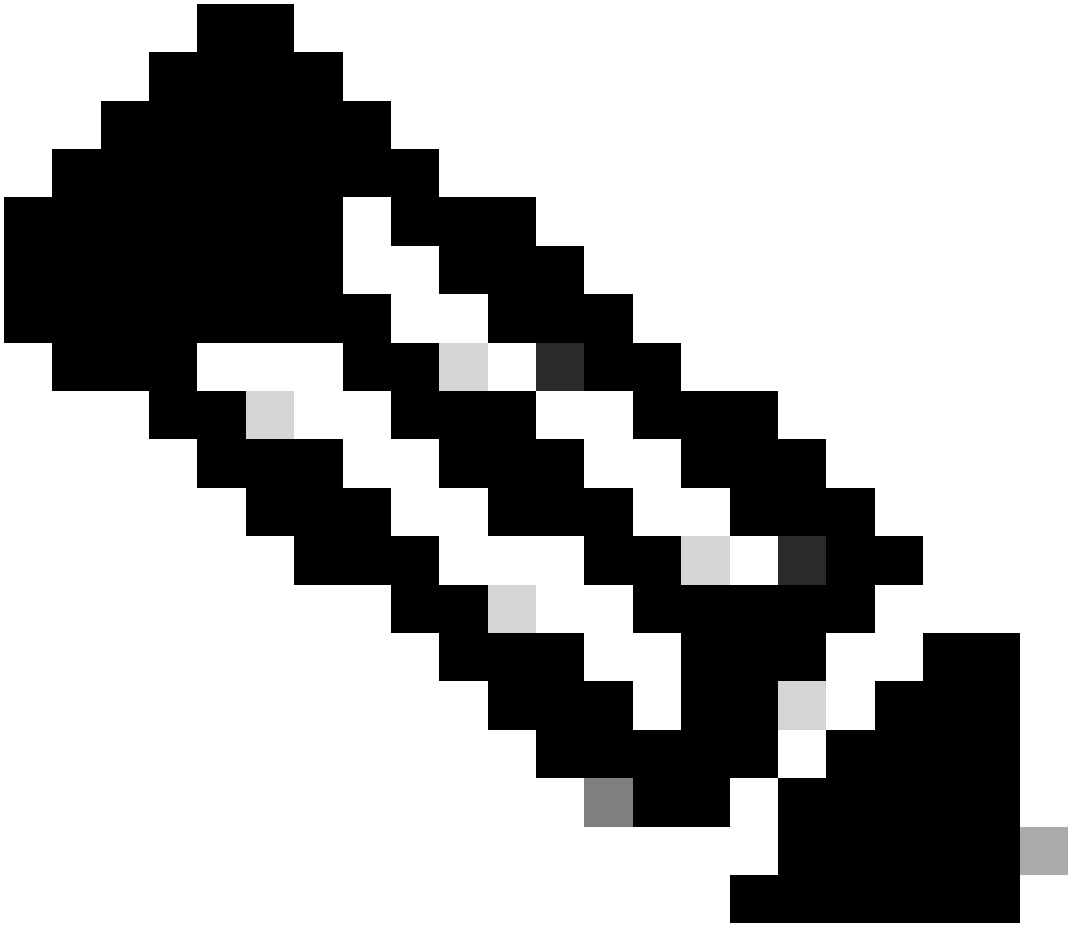
```
NAM - https://api.amp.cisco.com/v1/computers/
EU - https://api.eu.amp.cisco.com/v1/computers/
APJC - https://api.apjc.amp.cisco.com/v1/computers/
```

- قم بإنشاء ملف config.txt في نفس الدليل مثل النص التنفيذي مع المحتوى المذكور. مثال على ملف config.txt:

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

- قم بإنشاء ملف guids.txt في نفس الدليل مثل البرنامج النصي مع قائمة بمعرفات GUID للموصل، واحد لكل سطر. قم بإضافة معرفات عمومية (GUID) حسب الحاجة. مثال على ملف guids.txt:

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



ملاحظة: يمكنك تجميع معرفات عمومية لنقاط النهاية الخاصة بك إما من خلال API GET /v1/computers أو من وحدة تحكم نقطة النهاية الآمنة من Cisco من خلال الانتقال إلى الإدارة > أجهزة الكمبيوتر، وتوسيع الإدخال لنقطة نهاية معينة، ونسخ GUID للموصل.

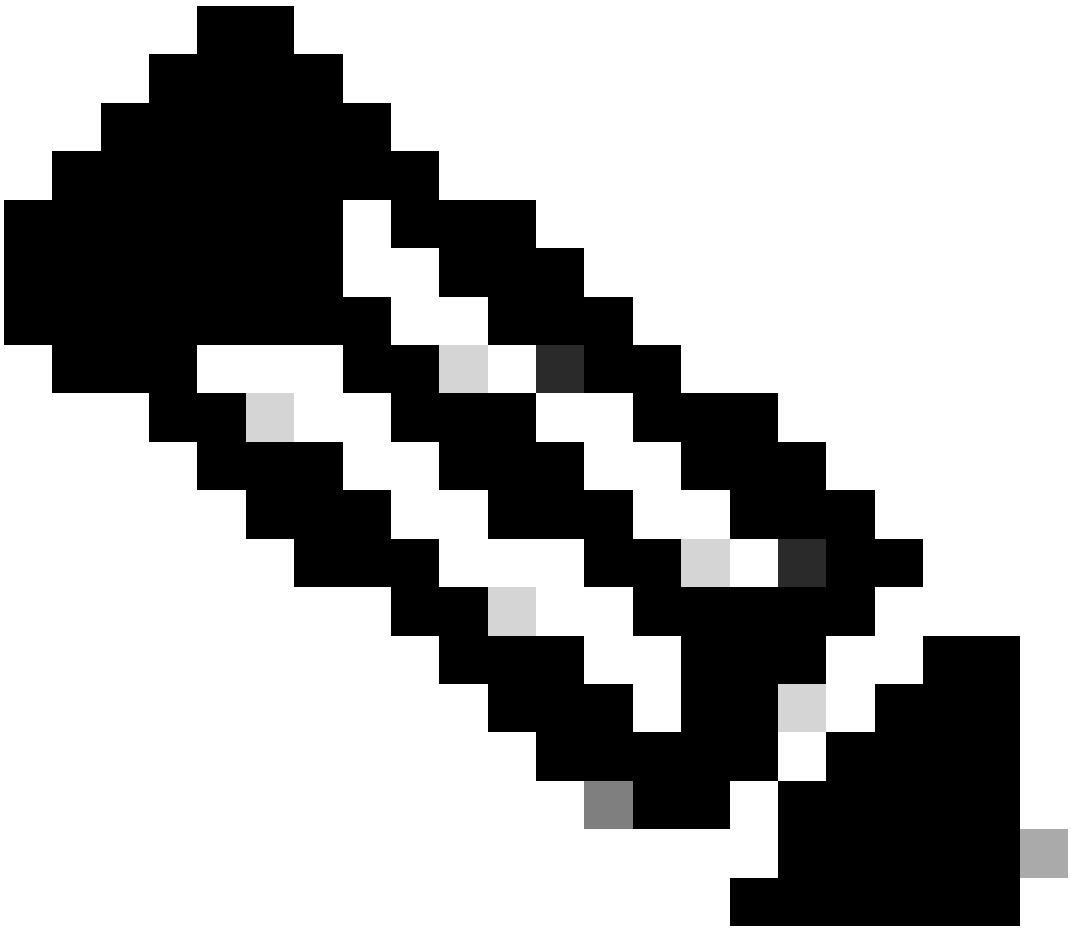
- فتح موجه أمر أو محطة طرفية. انتقل إلى الدليل الذي يوجد به start_isolation_script.py.
- قم بتنفيذ البرنامج النصي بتشغيل الأمر المذكور:

```
python start_isolation_script.py
```

التحقق من الصحة

- يحاول البرنامج النصي عزل كل نقطة نهاية محددة في ملف guids.txt.
- تحقق من الوحدة الطرفية أو موجه الأمر بحثا عن رسائل النجاح أو الخطأ لكل نقطة

نهاية.



ملاحظة: يمكن إستخدام الأمر script start_isolation.py المرفق لبدء العزل على نقاط النهاية، بينما تم تصميم stop_isolation.py لإيقاف العزل على نقاط النهاية. كل التعليمات الخاصة بتشغيل وتنفيذ النص التنفيذي تبقى كما هي.

حول هذه الترجمة
ترجمت Cisco هذا المستند باستخدام مجموعة من التقنيات الآلية والبشرية لتقديم محتوى دعم للمستخدمين في جميع أنحاء العالم بلغتهم الخاصة. يُرجى ملاحظة أن أفضل ترجمة آلية لن تكون دقيقة كما هو الحال مع الترجمة الاحترافية التي يقدمها مترجم محترف. تخلي Cisco Systems مسئوليتها عن دقة هذه الترجمات وتُوصي بالرجوع دائمًا إلى المستند الإنجليزي الأصلي (الرابط متوفر).