Automatizar el inicio/detención del aislamiento en varios terminales

Contenido

Introducción

Este documento describe cómo automatizar el aislamiento de parada/inicio en múltiples terminales usando la API para Cisco Secure Endpoint.

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- Cisco Secure Endpoint
- Cisco Secure Endpoint Console
- API de terminal seguro de Cisco
- Python

Componentes Utilizados

La información de este documento se basa en las siguientes versiones de software:

- Cisco Secure Endpoint 8.4.0.30201
- Entorno python de terminal a host
- Python 3.1.7

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si tiene una red en vivo,

asegúrese de entender el posible impacto de cualquier comando.

Antecedentes

- Se utiliza una solicitud PUT para iniciar el aislamiento.
- Se utiliza una solicitud DELETE para detener el aislamiento.
- Consulte la documentación de la API para obtener más información.

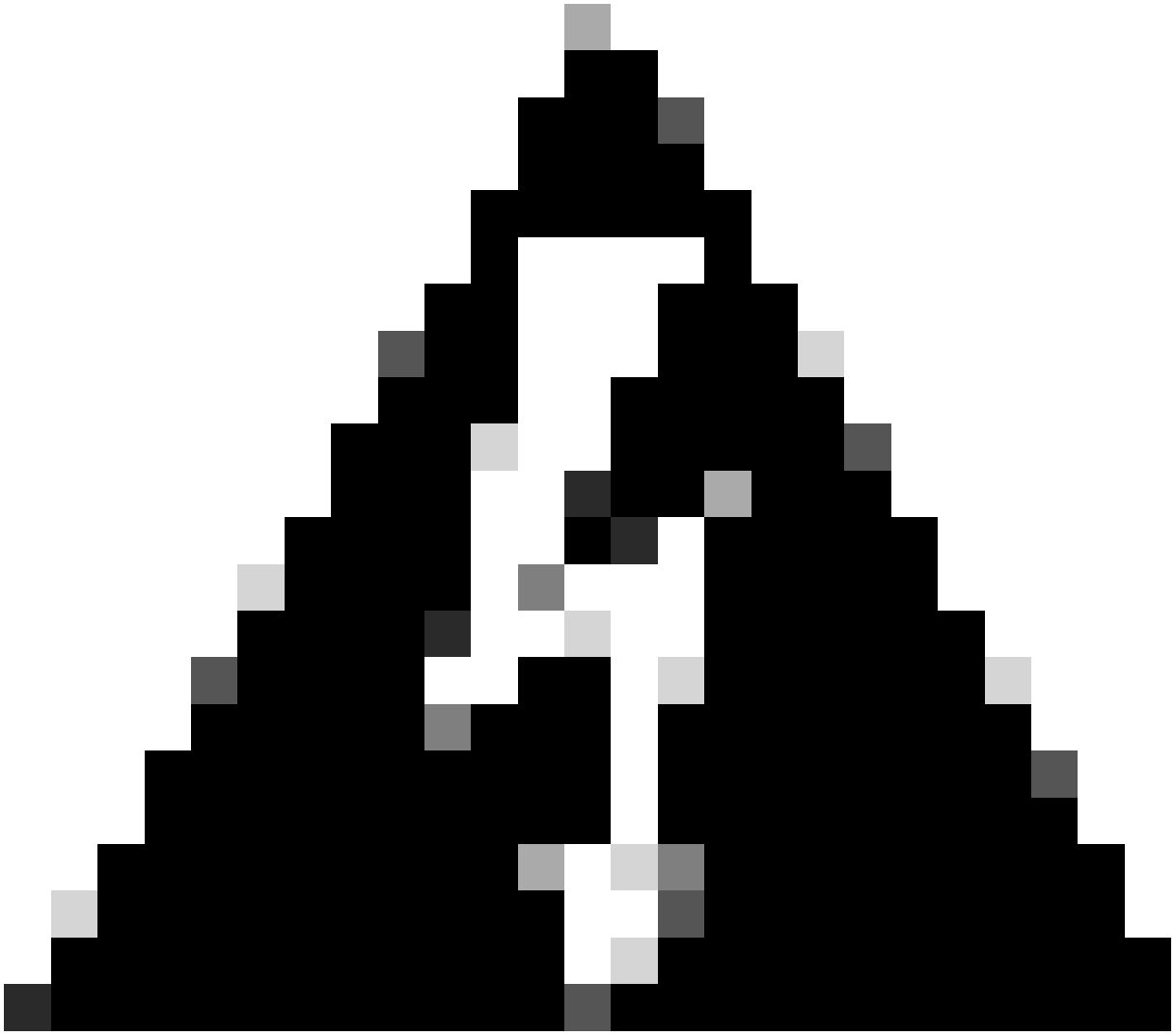
Problema

Cisco Secure Endpoint permite iniciar/detener el aislamiento en un equipo a la vez. Sin embargo, durante un incidente de seguridad, a menudo es necesario realizar estas operaciones en varios terminales simultáneamente para contener de forma eficaz las amenazas potenciales. La automatización del proceso de inicio/detención del aislamiento para terminales masivos mediante la API puede mejorar significativamente la eficacia de la respuesta ante incidentes y reducir el riesgo general para la red.

Solución

- La secuencia de comandos de Python proporcionada en este artículo se puede utilizar para iniciar o finalizar el aislamiento en varios terminales de su organización mediante las credenciales de la API de terminales seguros.
- Para generar las credenciales de la API de AMP, consulte Descripción general de la API de Cisco AMP para terminales
- Para utilizar la secuencia de comandos proporcionada, debe instalar pythonen los terminales.
- Después de instalar python, instale el módulo de solicitudes

```
pip install requests
```



Advertencia: la secuencia de comandos se proporciona únicamente con fines ilustrativos y tiene como objetivo demostrar cómo automatizar la función de aislamiento de terminales mediante la API. El centro de asistencia técnica Cisco Technical Assistance Center (TAC) no está involucrado en la resolución de problemas relacionados con este script. Los usuarios deben tener cuidado y probar exhaustivamente la secuencia de comandos en un entorno seguro antes de implementarla en una configuración de producción.

Guión

Puede utilizar la secuencia de comandos proporcionada para iniciar el aislamiento en varios terminales de su empresa:

```
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)
```

Instrucciones

- Para generar las credenciales de la API de AMP, consulte Descripción general de la API de Cisco AMP para terminales
- Utilice BASE_URL mencionado para su región:

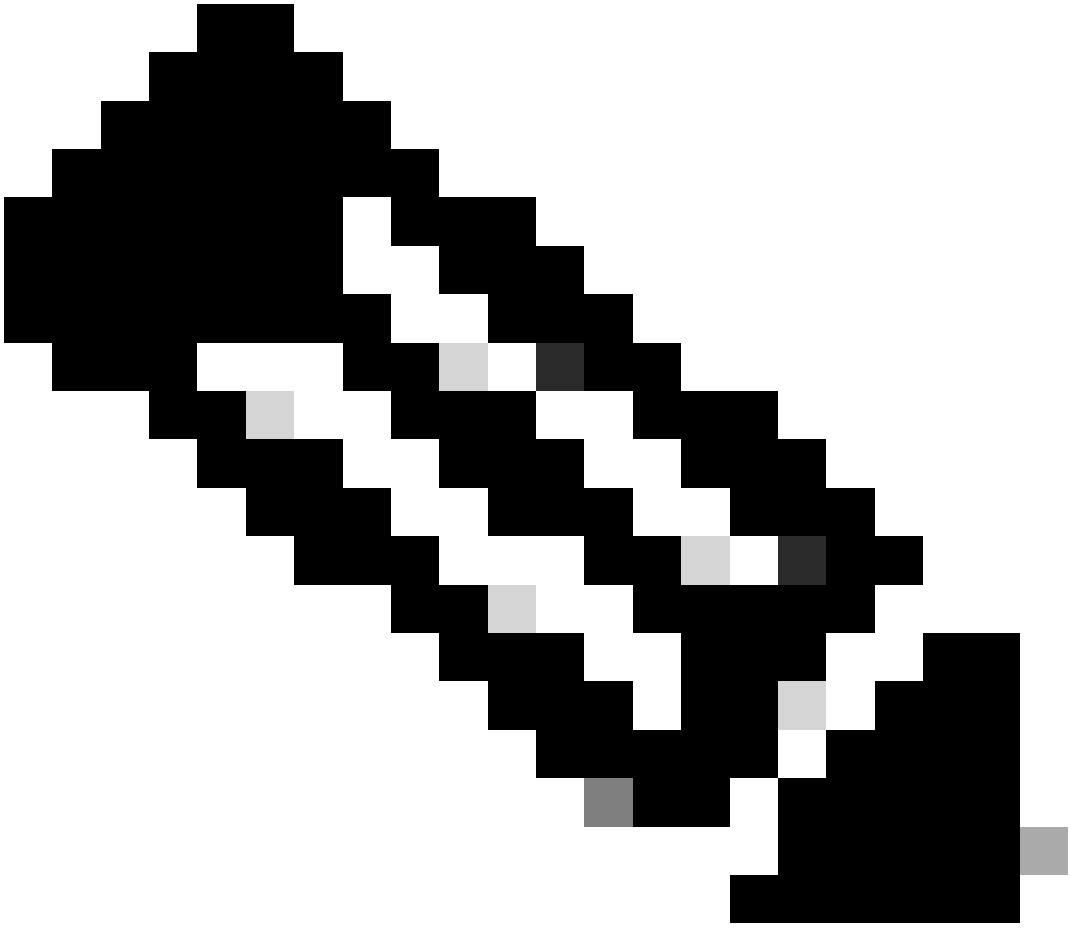
```
NAM - https://api.amp.cisco.com/v1/computers/
EU - https://api.eu.amp.cisco.com/v1/computers/
APJC - https://api.apjc.amp.cisco.com/v1/computers/
```

- Cree un archivo config.txt en el mismo directorio que la secuencia de comandos con el contenido mencionado. Ejemplo de archivo config.txt:

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

- Cree un archivo guids.txt en el mismo directorio que la secuencia de comandos con la lista de GUID de conector, uno por línea. Agregue tantos GUID como sea necesario. Ejemplo de archivo guids.txt:

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



Nota: Puede recopilar los GUID de sus extremos a través de la API GET /v1/computers o desde Cisco Secure Endpoint Console navegando hasta Management > Computers, expandiendo la entrada para un extremo específico y copiando el GUID del conector.

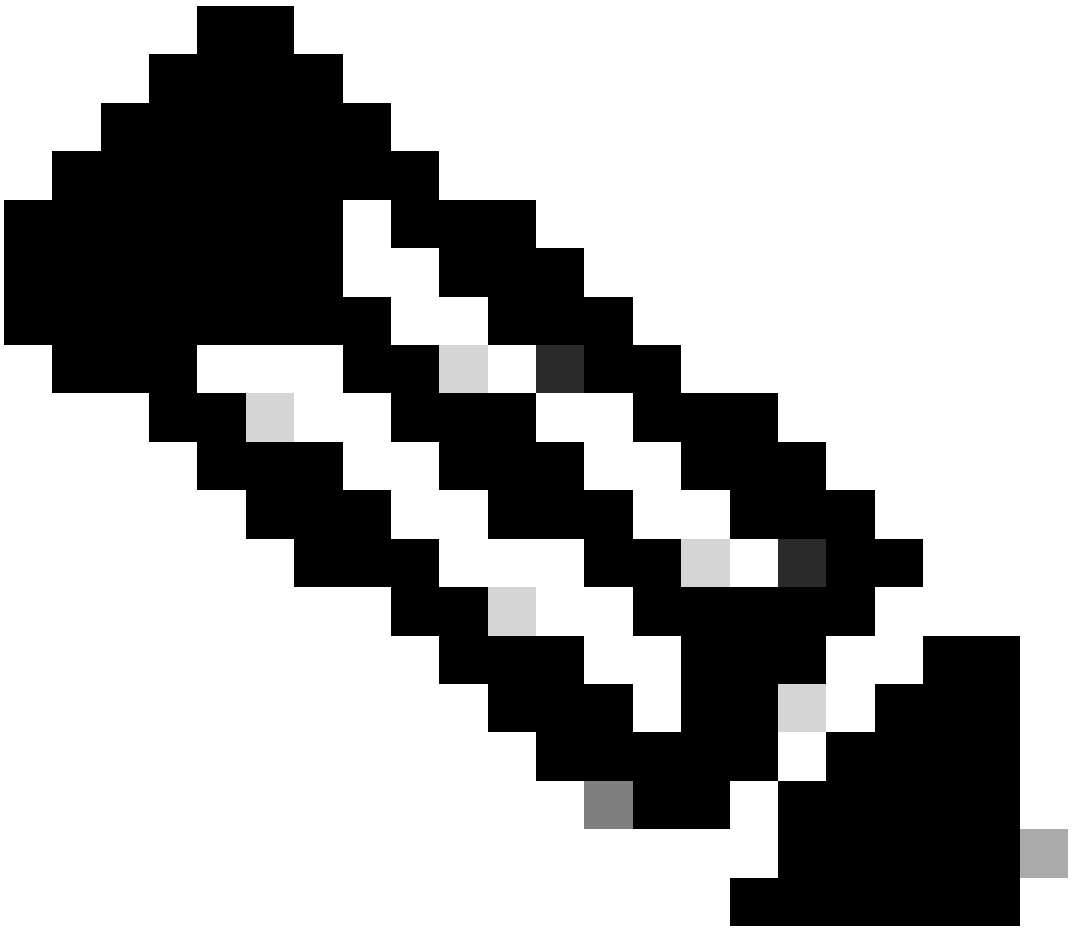
- Abra un terminal o símbolo del sistema. Navegue hasta el directorio donde se encuentra start_isolation_script.py.
- Ejecute el script ejecutando el comando mencionado:

```
python start_isolation_script.py
```

Verificación

- La secuencia de comandos intenta aislar cada extremo especificado en el archivo guids.txt.
- Verifique el terminal o el símbolo del sistema para ver si hay mensajes de éxito o error para

cada punto final.



Nota: El script adjunto start_isolation.py se puede utilizar para iniciar el aislamiento en los terminales, mientras que stop_isolation.py está diseñado para detener el aislamiento en los terminales. Todas las instrucciones para ejecutar y ejecutar la secuencia de comandos siguen siendo las mismas.

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).