Enviar objetos em massa para o FMC usando REST-API

Contents

Introduction

Este documento descreve como um administrador da API (Application Programming Interface, Interface de programação de aplicativos) pode enviar Objetos de rede, porta e URL em massa para o Firepower Management Center (FMC).

Prerequisites

Requirements

A Cisco recomenda que você tenha conhecimento destes tópicos:

- Entendendo várias chamadas de API REST. (O que são APIs REST?)
- Revisão do Guia de Início Rápido da API do FMC
- Revisão dos objetos reutilizáveis da FMC
- Conhecimento básico da biblioteca de pedidos Python

Componentes Utilizados

- Firepower Management Center que suporta APIs REST (versão 6.1 ou superior) com API REST habilitada
- Interações de API REST usando Python.

Limitações

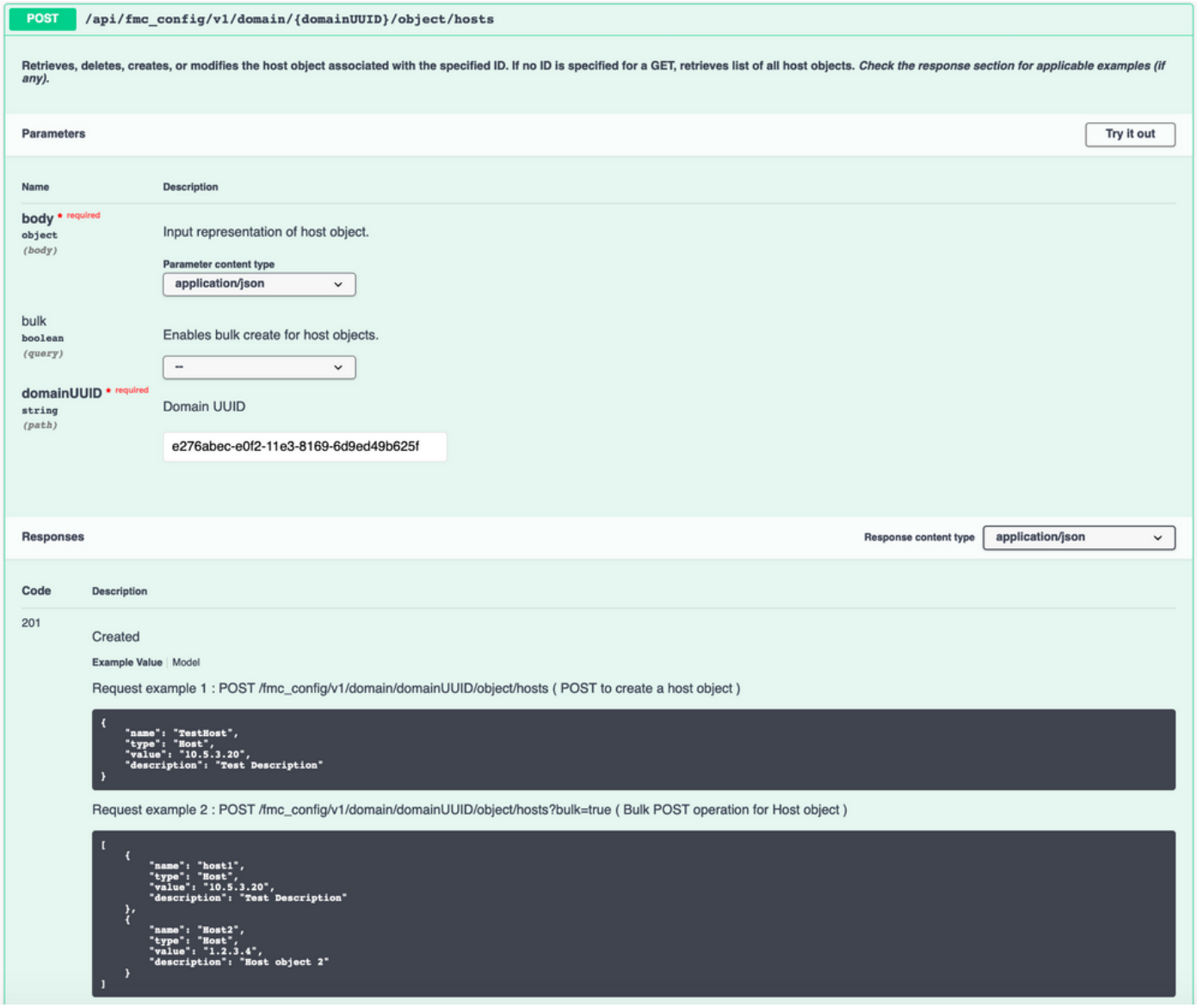
- O FMC não aceita que o nome do objeto tenha mais de 64 caracteres.
- O nome do objeto não deve ter espaço no início do nome do objeto e ponto e vírgula no final.
- O payload não pode conter mais de 1.000 entradas em um único Bulk Push.
- O tamanho da carga útil não pode ser superior a 2 MB em um único envio em massa.

Informações de Apoio

As APIs REST são cada vez mais populares devido à abordagem programável leve que os gerentes de rede podem usar para configurar e gerenciar suas redes. O FMC suporta configuração e gerenciamento usando qualquer cliente REST e também usando o explorador de API interno.

O exemplo neste documento pega um arquivo CSV como uma entrada e envia os objetos para o FMC através da interface da API REST. O documento aborda apenas o envio em massa de rede do host e uma lógica semelhante pode ser estendida para os outros objetos. Um código de exemplo é anexado ao documento para objetos de URL e porta.

Esta é a referência de API para o POST em hosts de rede usados, como mostrado na imagem:



Configurar

Etapa 1. Ative a API REST e gere o Token de autenticação. Para ver as etapas detalhadas de configuração e exemplos, consulte Gerar Token de Autenticação no FMC.

```
import requests import csv import json from requests.auth import HTTPBasicAuth from getpass
import getpass address = input("Enter IP Address of the FMC: ") username = input ("Enter
Username: ") password = getpass("Enter Password: ") api_uri =
"/api/fmc_platform/v1/auth/generatetoken" url = "https://" + address + api_uri response =
requests.request("POST", url, verify=False, auth=HTTPBasicAuth(username, password)) accesstoken
= response.headers["X-auth-access-token"] refreshtoken = response.headers["X-auth-refresh-
token"] DOMAIN_UUID = response.headers["DOMAIN_UUID"]
```

Etapa 2. Converter o arquivo CSV fornecido em um dicionário para ser usado como payload JSON para a solicitação. O arquivo CSV de exemplo para cada tipo de objeto está anexado ao documento.

	A	B	C	D	
1	name	description	type	value	
2	Host-1	Host-1	Host	10.10.10.10	
3	Host-2	Host-2	Host	10.10.10.1	
4	Network-1	Network-1	Network	10.10.9.0/24	
5	Host-3	Host-3	Host	10.10.10.2	
6	Range-1	Rannge-1	Range	10.20.20.1-10.20.20.20	
7					

```
csvFilePath = input("Please enter the CSV Filepath (For eg. : path/to/file/objects.csv) :") host
= [] with open(csvFilePath, encoding='utf-8-sig') as csvf: csvReader = csv.DictReader(csvf) for
rows in csvReader: if rows['type'] == "Host": host.append(rows) host_payload = json.dumps(host)
```

O host_payload neste estágio é igual ao mostrado na imagem:

```
[{ "name": "Host-1", "description": "Host-1", "type": "Host", "value": "10.10.10.10" }, {
"name": "Host-2", "description": "Host-2", "type": "Host", "value": "10.10.10.1" }, { "name":
"Host-3", "description": "Host-3", "type": "Host", "value": "10.10.10.2" } ]
```

Etapa 3. Crie a solicitação da entrada recebida das etapas anteriores e envie a solicitação se o payload não estiver vazio.

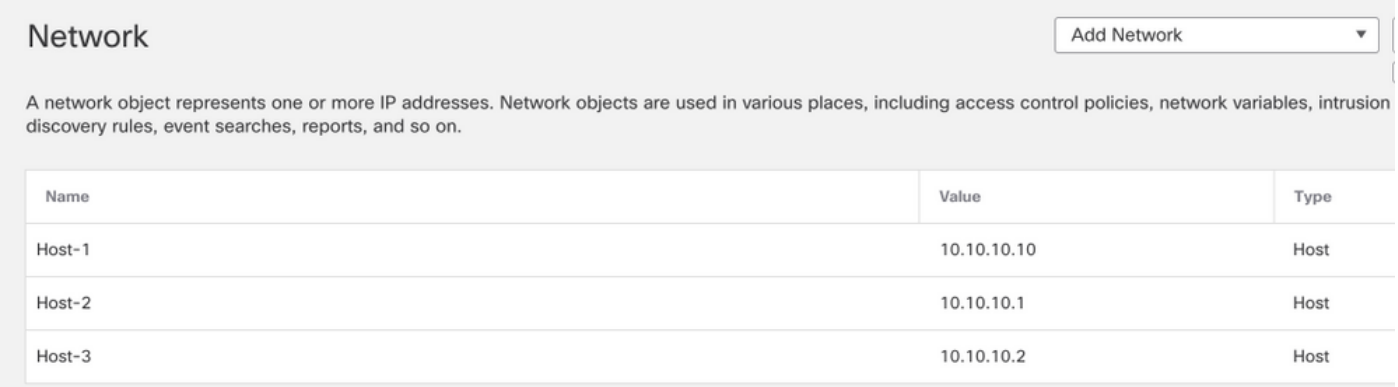
```
host_api_uri = "/api/fmc_config/v1/domain/" + DOMAIN_UUID + "/object/hosts?bulk =true" host_url
= "https://" + address + host_api_uri headers = { 'Content-Type': 'application/json', 'x-auth-
access-token': accesstoken } if host != []: response = requests.request("POST", host_url,
headers=headers, data = host_payload, verify = False) else : print("Please Validate that the CSV
file provided is correct or at correct location")
```

Verificar

- Imprima o código de status da resposta para verificar se a solicitação foi bem-sucedida ou falhou, como mostrado aqui.

```
if response.status_code == 201 or response.status_code == 202: print("Host Objects successfully
pushed") else: print("Host Object creation failed")
```

- Faça login no FMC Navegue até **Object > Object Management > Network** e verifique os Host Objects, como mostrado na imagem:



Troubleshoot

- Ao usar o cliente REST, você pode ver erros relacionados ao problema de certificado SSL devido a um certificado autoassinado. Você pode desativar essa validação dependendo do cliente que está usando.
- Os tokens de autenticação da API REST do FMC são válidos por 30 minutos e podem ser atualizados até três vezes.
- O erro relacionado à solicitação pode ser extraído do corpo da resposta. Isso pode ser coletado como um arquivo de log para ajudar na solução de problemas.

```
logfile = "requestlog.txt" log = open(logfile,"w+") log.write(response.text) log.close
```

- Todas as solicitações REST são registradas nesses dois arquivos de log no FMC. Procure seu URL (ex. .../object/hosts) com a operação correta(Se você estiver procurando um erro para a operação GET, certifique-se de que seu log inicie algo como GET ...objeto/hosts)

```
tail -f /var/opt/CSCOpx/MDC/tomcat/logs/stdout.logs tail -f
/var/opt/CSCOpx/MDC/log/operation/usmsharedsvcs.log
```