PythonおよびREST APIを使用したセキュアアクセス宛先リストの管理

内容

スポイラー（参照用に強調表示）
シスコでは、この開発設計を公式にサポートしておりません。これは、APIがアプリケーションとインターフェイスする方法を理解しやすくするための参照例としてのみ意図されています。ユーザは、この設計を教育目的でのみ使用し、実稼働レベルの実装の基盤として使用することはできません。 この文書に記載されているコードの実行は、お客様の責任において行われるものであり、シスコはその使用から生じるいかなる問題に対しても責任を負いません。

シスコでは、この開発設計を公式にサポートしておりません。これは、APIがアプリケーションとインターフェイスする方法を理解しやすくするための参照例としてのみ意図されています。ユーザは、この設計を教育目的でのみ使用し、実稼働レベルの実装の基盤として使用することはできません。 この文書に記載されているコードの実行は、お客様の責任において行われるものであり、シスコはその使用から生じるいかなる問題に対しても責任を負いません。

はじめに

このドキュメントでは、PythonおよびREST APIを使用して、宛先リストで可能なすべての操作を実行する方法について説明します。

前提条件

次の項目に関する知識があることが推奨されます。

1. Python
2. REST API
3. シスコセキュアアクセス

要件

次の手順に進む前に、次の要件を満たす必要があります。

- Full Adminuserロールを持つCisco Secure Accessユーザアカウント。
- セキュアアクセスにサインインするためのCisco Security Cloudシングルサインオン(SCSO)アカウント
- セキュアアクセスAPIキーの作成

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- セキュアアクセスダッシュボード
- Python 3.x

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

設定

エラー処理、トークンの有効性（3600秒）など、複数の側面を考慮してこのコードを記述する方法は複数あります。

スクリプトを実行する前に、次のPythonライブラリがインストールされていることを確認してください。

```
pip install requests
pip install oauthlib
pip install requests_oauthlib
```

スクリプト

このスクリプトでは、client_idとclient_secretを、それぞれAPI KeyとKey Secretに置き換えてください。

```
from oauthlib.oauth2 import BackendApplicationClient
from oauthlib.oauth2 import TokenExpiredError
from requests_oauthlib import OAuth2Session
from requests.auth import HTTPBasicAuth
import time
import requests
import pprint
import json


def fetch_headers(BToken):
    BT = f"Bearer {BToken}"
    headers = { 'Authorization':BT,
            "Content-Type": "application/json",
```

```
            "Accept": "application/json"
            }
    return headers

# GET OAUTH 2.0 TOKEN
def getToken():
    token_url = 'https://api.sse.cisco.com/auth/v2/token'
    try:
        #ASSIGN your API Key to the variable client_id and Secret Key to the variable client_secret
        client_id = "XXXXXXXXXXXXXXXXXXXXXXXXXXXX"
        client_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXX"

        auth = HTTPBasicAuth(client_id, client_secret)
        client = BackendApplicationClient(client_id=client_id)
        oauth = OAuth2Session(client=client)
        token = oauth.fetch_token(token_url=token_url, auth=auth)
        print("\n######Token Generated Successfully######\n")
        return token
    except e as Exception:
        print(f"Encountered an error while Fetching the TOKEN :: {e}")


# 1 - GET DESTINATION LISTS
def fetch_destinationlists(h):
    url = "https://api.sse.cisco.com/policies/v2/destinationlists"
    try:
        response = requests.request('GET', url, headers=h)
        json_object = json.loads(response.content)

        #pprint.pprint(json_object)
        x=1
        for item in json_object["data"]:
            print(f"Destination List : {x}")
            pprint.pprint(f"Name : {item['name']}")
            pprint.pprint(f"ID : {item['id']}")
            #pprint.pprint(f"Destination Count : {item['meta']['destinationCount']}")
            print("\n")
            x+=1
    except e as Exception:
        print(f"Encountered an Error while Fetching the Destination Lists :: {e}")


# 2 - GET DESTINATION LIST
def get_destinationlist(h):
    try:
        choice = input("Enter the ID of the DestinationList:: ")
        url = "https://api.sse.cisco.com/policies/v2/destinationlists/" + choice
        response = requests.request('GET', url, headers=h)
        json_object = json.loads(response.content)
        print("\n\n")
        pprint.pprint(json_object)
        print("\n\n")
    except e as Exception:
        print(f"Encountered an Error while Fetching the Destination List Details :: {e}")


# 3 - CREATE DESTINATION LIST
def create_destinationlist(h):
    url = "https://api.sse.cisco.com/policies/v2/destinationlists"
    try:
        naav = input("Name of the DestinationList :: ")
        payload = {
```

```
            "access": "none",
            "isGlobal": False,
            "name": naav,
            }
        response = requests.request('POST', url, headers=h, data = json.dumps(payload))
        json_object = json.loads(response.content)
        print("\n\n")
        pprint.pprint(json_object)
        print("\n\n")
    except e as Exception:
        print(f"Encountered an Error while Creating the Destination List :: {e}")


# 4 - UPDATE DESTINATION LIST NAME
def patch_destinationlist(h):
    try:
        choice = input("Enter the ID of the DestinationList for changing it's name :: ")
        url = "https://api.sse.cisco.com/policies/v2/destinationlists/" + choice
        naav = input("Enter New Name :: ")
        payload = {"name": naav}

        response = requests.request('PATCH', url, headers=h, data = json.dumps(payload))
        json_object = json.loads(response.content)
        print("\n\n")
        pprint.pprint(json_object)
        print("\n\n")
    except e as Exception:
        print(f"Encountered an Error while Updating the Destination List :: {e}")


# 5 - DELETE DESTINATION LIST
def delete_destinationlist(h):
    try:
        choice = input("Enter the ID of the DestinationList for DELETION :: ")
        url = "https://api.sse.cisco.com/policies/v2/destinationlists/" + choice

        response = requests.request('DELETE', url, headers=h)
        json_object = json.loads(response.content)
        print("\n\n")
        pprint.pprint(json_object)
        print("\n\n")
    except Exception as e:
        print(f"Encountered an Error while Deleting the Destination List :: {e}")


# 6 - GET DESTINATIONS FROM A DESTINATION LIST
def fetch_detail(h):
    try:
        choice = input("DestinationList ID: ")

        url2 = "https://api.sse.cisco.com/policies/v2/destinationlists/" + choice + "/destinations"

        response = requests.request('GET', url2, headers=h)
        print("\n")
        json_dest = json.loads(response.content)
        pprint.pprint(json_dest)
        print("\n\n")
    except e as Exception:
        print(f"Encountered an Error while Fetching the Destinations from the Destination List :: {e}")


# 7 - ADD DESTINATIONS TO A DESTINATION LIST
```

```
def add_destinations(h):
    try:
        choice = input("Enter the ID of the DestinationList :: ")
        url = "https://api.sse.cisco.com/policies/v2/destinationlists/" + choice + "/destinations"
        destination_to_add = input("\nEnter the destination that you want to add :: ")
        payload = [{"destination": destination_to_add}]

        response = requests.request('POST', url, headers=h, data = json.dumps(payload))
        print("\n")
        json_dest = json.loads(response.content)
        pprint.pprint(json_dest)
        print("\n\n")
    except e as Exception:
        print(f"Encountered an Error while Adding the Destination to the Destination List :: {e}")


# 8 - DELETE DESTINATIONS FROM DESTINATION LIST
def delete_entry(h):
    try:
        choice_del = input("\nCONFIRM DestinationList ID from which you want to delete the Destination
        url3 = "https://api.sse.cisco.com/policies/v2/destinationlists/" + choice_del + "/destinations/
        dest = int(input("ID of the Destination that you want to remove: "))

        payload = f"[{dest}]"
        response = requests.request('DELETE', url3, headers=h, data=payload)
        json_del = json.loads(response.content)
        print("\n\n")
        pprint.pprint(json_del)
        print("\n\n")
    except e as Exception:
        print(f"Encountered an Error while Deleting a Destination from the Destination List :: {e}")



#FETCH COOKIE
possess_cookie = " "
while possess_cookie not in ["Y","y","N","n"]:
    possess_cookie = input("Token Already Generated? (Y/N) :: ")
    if possess_cookie.upper() =="N":
        cook = getToken()
        with open("cookie.txt","w") as wr:
            wr.writelines(cook["access_token"])
#         print(f"Access Token = {cook["access_token"]}")


#FETCH HEADERS
with open("cookie.txt","r") as ree:
    h = fetch_headers(ree.readline())


print("\n")
while True:
    action = input("""Available operations:
1. Get Destination Lists
2. Get Destination List
3. Create Destination List
4. Update Destination List Name
5. Delete Destination List
6. Get Destinations from Destination List
7. Add Destinations to a Destination List
8. Delete Destinations from Destination List
9. Exit
```

```
Enter Your Choice :: """)

    print("\n")
    operation = action.replace(" ","")
    if operation == "1":
        fetch_destinationlists(h)

    elif operation == "2":
        fetch_destinationlists(h)
        get_destinationlist(h)

    elif operation == "3":
        create_destinationlist(h)

    elif operation == "4":
        fetch_destinationlists(h)
        patch_destinationlist(h)

    elif operation == "5":
        fetch_destinationlists(h)
        delete_destinationlist(h)

    elif operation == "6":
        fetch_destinationlists(h)
        fetch_detail(h)

    elif operation == "7":
        fetch_destinationlists(h)
        add_destinations(h)

    elif operation == "8":
        fetch_destinationlists(h)
        fetch_detail(h)
        delete_entry(h)

    elif operation == "9":
        break

    else:
        print("\n")
        print("==========INCORRECT INPUT==========")
        print("\n")

print("Thank You!!! Good Bye!!!")
time.sleep(5)
```

出力：

このスクリプトからの出力は、次のようになります。

```
Token Already Generated? (Y/N) :: y

Available operations:
1. Get Destination Lists
2. Get Destination List
3. Create Destination List
```

```
4. Update Destination List Name
5. Delete Destination List
6. Get Destinations from Destination List
7. Add Destinations to a Destination List
8. Delete Destinations from Destination List
9. Exit

Enter Your Choice :: 1
```

このプログラムが正常に実行されると、最初にCookieに関する質問が表示されます。Cookie Already Generated? (Y/N)この質問を行う理由は、一度cookieが生成されると3600秒（1時間）有効になるため、cookieを複数回生成しないようにするためです。 yまたはYと入力した場合、新しいcookieは生成されません。ただし、nまたはNと入力すると、新しいcookieが生成され、同じディレクトリ/フォルダ内のローカルテキストファイルに保存されます。このファイルのcookieは、以降の要求で使用されます。

Errors

このエラーは、DestinationList IDを指定する必要がある操作に対して誤ったIDを入力した場合に発生することがあります。

```
{'message': 'no Route matched with those values'}
```

DestinationListの作成中に、255文字を超えるDestinationListの名前を指定すると、次のエラーが表示されます。

```
{'code': 400,
'code_text': 'Bad Request',
'error': 'invalid_request',
'message': {'name': {'code': 'string_invalid',
'code_text': 'must be fewer than 255 characters long'}},
'statusCode': 400,
'txId': 'l23f0cjk62fe'}
```

また、『セキュアアクセス開発者ユーザガイド』を使用して、ポリシー、ローミングコンピュータ、レポートなどに関する情報を取得することもできます。

トラブルシュート

Secure Access APIエンドポイントは、HTTP応答コードを使用してAPI要求の成功または失敗を示します。一般に、2xxの範囲のコードは成功を示し、4xxの範囲のコードは提供された情報に起因するエラーを示し、5xxの範囲のコードはサーバエラーを示します。問題を解決するアプローチは、受信した応答コードによって異なります。

200	**OK**	Success. Everything worked as expected.
201	**Created**	New resource created.
202	**Accepted**	Success. Action is queued.
204	**No Content**	Success. Response with no message body.
400	**Bad Request**	Likely missing a required parameter or malformed JSON. The syntax of your query may need to be revised. Check for any spaces preceding, trailing, or in the domain name of the domain you are trying to query.
401	**Unauthorized**	The authorization header is missing or the key and secret pair is invalid. Ensure your API token is valid.
403	**Forbidden**	The client is unauthorized to access the content.
404	**Not Found**	The requested resource doesn't exist. Check the syntax of your query or ensure the IP and domain are valid.
409	**Conflict**	The client requests that the server create the resource, but the resource already exists in the collection.
429	**Exceeded Limit**	Too many requests received in a given amount of time. You may have exceeded the rate limits for your organization or package.
413	**Content Too Large**	The request payload is larger than the limits defined by the server.

REST API：応答コード1

500	**Internal Server Error**	Something wrong with the server.
503	**Service Unavailable**	Server is unable to complete request.

REST API：応答コード2

関連情報

- Cisco Secure Accessユーザガイド
- シスコテクニカルサポートおよびダウンロード
- セキュアアクセスAPIキーの追加
- 開発者ユーザガイド
- PythonでREST APIを使用するためのセキュアアクセスの設定
- cURLを使用した通知先リストの管理

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。