

# Configurer Postman pour exécuter des API sur vManage

## Table des matières

[Introduction](#)

[Configuration système nécessaire](#)

[Informations générales](#)

[Configurer Postman pour exécuter les API](#)

[Étape 1. Ouvrez Postman et créez une nouvelle requête HTTP.](#)

[Étape 2. Authentifiez-vous avec votre nom d'utilisateur et votre mot de passe pour vManage.](#)

[Étape 3. Demander un jeton](#)

[Étape 4. Exécutez une autre API pour vManage.](#)

[Étape 5. Fermez votre session](#)

[Exécuter des appels API dans un environnement automatisé](#)

[Comment enregistrer un jeton dans une variable ?](#)

[Comment effacer le cookie SESSIONID pour les nouvelles sessions ?](#)

[Comment utiliser Collection Runner ?](#)

## Introduction

Ce document décrit comment exécuter des API (Application Programming Interfaces) avec Postman.

## Configuration système nécessaire

- Facteur installé
- Accès à vManage et identifiants de nom d'utilisateur et de mot de passe

**Remarque** : si vous n'avez pas Postman, téléchargez-le sur <https://www.postman.com/downloads/>

## Informations générales

Les verbes HTTP principaux ou les plus couramment utilisés (ou méthodes, comme on les appelle) sont POST, GET, PUT, PATCH et DELETE.

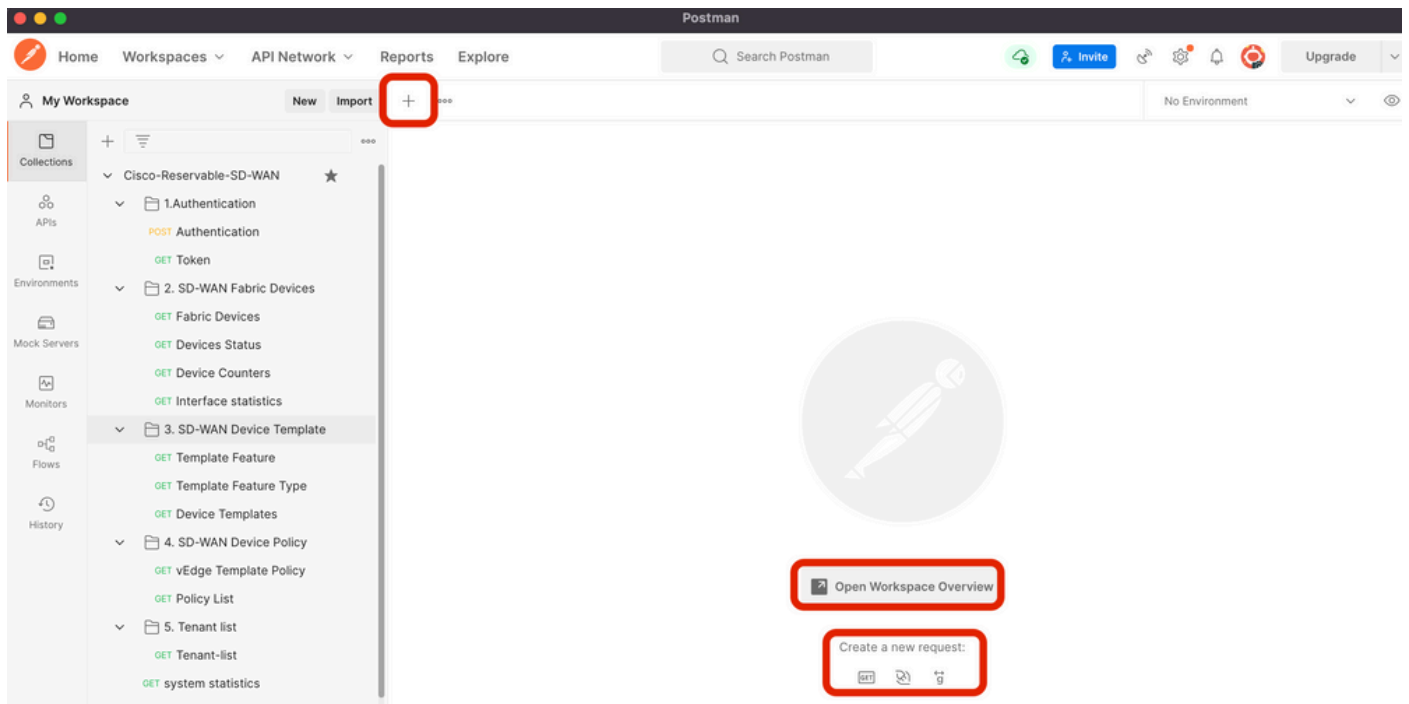
Elles correspondent respectivement aux opérations de création, de lecture, de mise à jour et de suppression (ou CRUD).

Il y a aussi un certain nombre d'autres verbes, mais ils sont utilisés moins fréquemment. Parmi ces méthodes moins fréquentes, OPTIONS et HEAD sont utilisées plus souvent que d'autres.

## Configurer Postman pour exécuter les API

## Étape 1. Ouvrez Postman et créez une nouvelle requête HTTP.

Vous pouvez créer de nouvelles requêtes HTTP si vous cliquez sur l'une des options mises en surbrillance.



Créez une nouvelle requête HTTP.

## Étape 2. Authentifiez-vous avec votre nom d'utilisateur et votre mot de passe pour vManage.

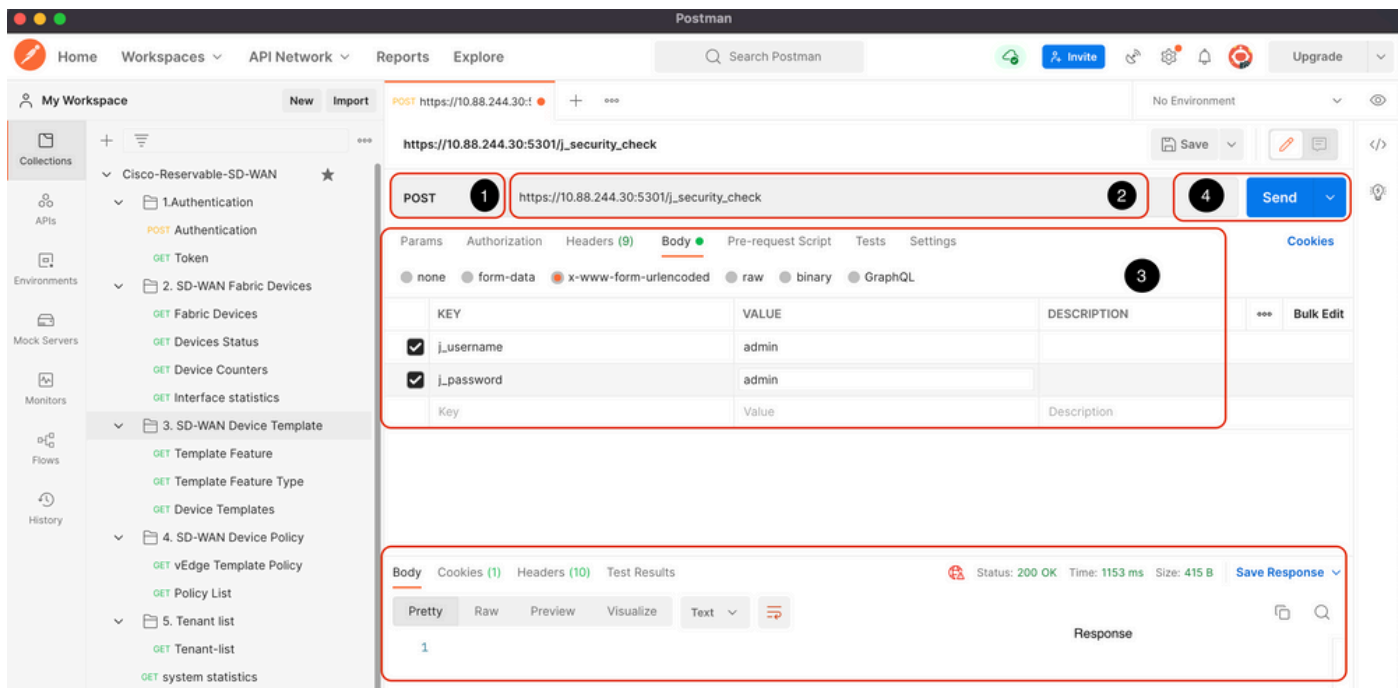
Créez une autre requête HTTP.

1. Sélectionnez **POST** comme verbe HTTP.
2. Ajoutez [https://<vmanage-ip>/j\\_security\\_checknext](https://<vmanage-ip>/j_security_checknext) à POST.
3. Cliquez sur **Body** et ajoutez en tant que **KEY** les paramètres **j\_username** et **j\_password** et leurs valeurs respectivement.
4. Cliquez sur **Envoyer**.

**Remarque** : dans cet exemple, l'adresse IP vManage est 10.88.244.30 et le port est 5301

**Remarque** : en tant que valeurs de nom d'utilisateur et de mot de passe, nous utilisons admin.

Complétez les paramètres dans Postman.



Authentification vManage.

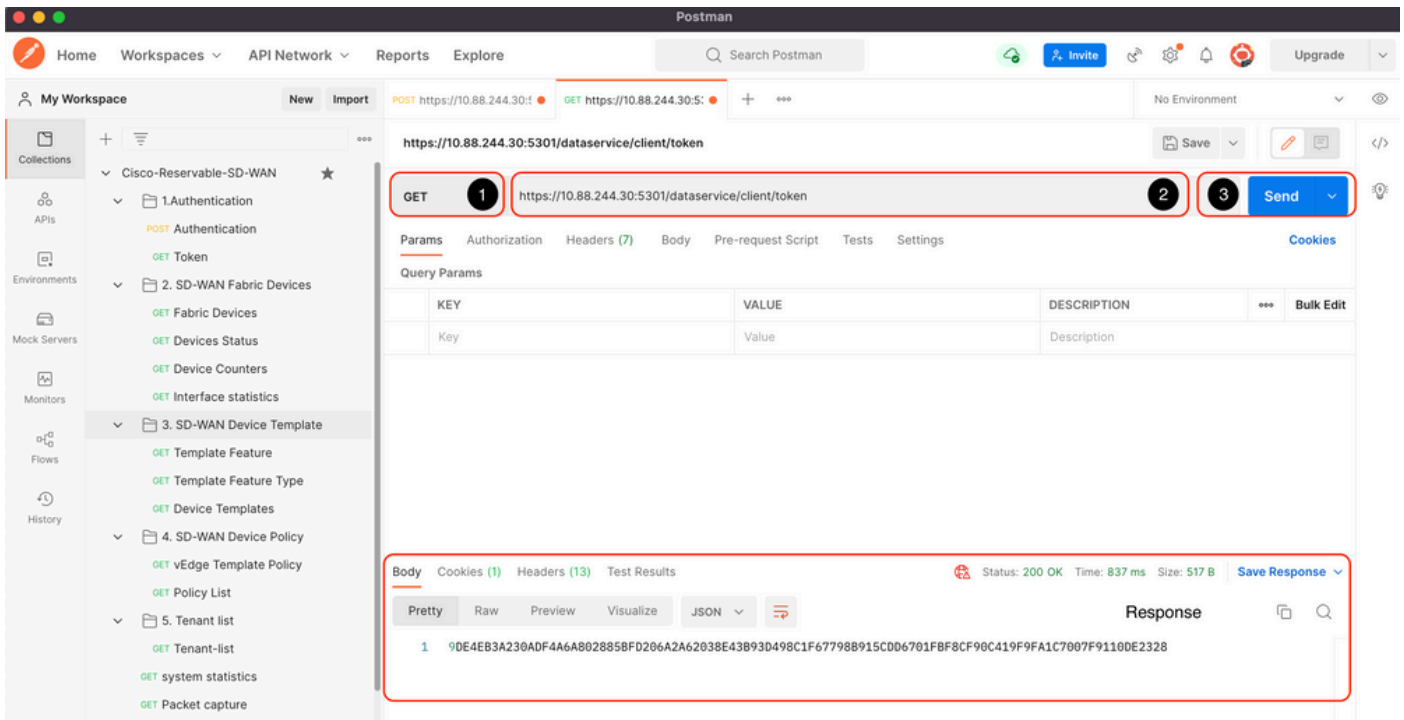
**Attention :** la réponse de cet appel API doit être vide

### Étape 3. Demander un jeton

1. Sélectionnez **GET** comme verbe HTTP.
2. Ajoutez les détails de l'appel API en regard de GET <https://<vmanage-ip>/dataservice/client/token>
3. Cliquez sur **Envoyer**

**Remarque :** depuis la version 19.2.1 de vManage, il est obligatoire qu'un utilisateur correctement connecté envoie un jeton X-XSRG-TOKEN ou CSRF pour chaque opération POST/PUT/DELETE via un appel d'API.

Une fois l'appel d'API exécuté, vous obtenez une chaîne de réponse dans le corps. Enregistrez cette chaîne. L'image illustrée illustre la sortie In Postman.



*Demander un jeton pour vManage*

**Avertissement :** si vous n'avez pas reçu de jeton, comme illustré dans l'image, répétez l'étape.

## Étape 4. Exécutez une autre API pour vManage.

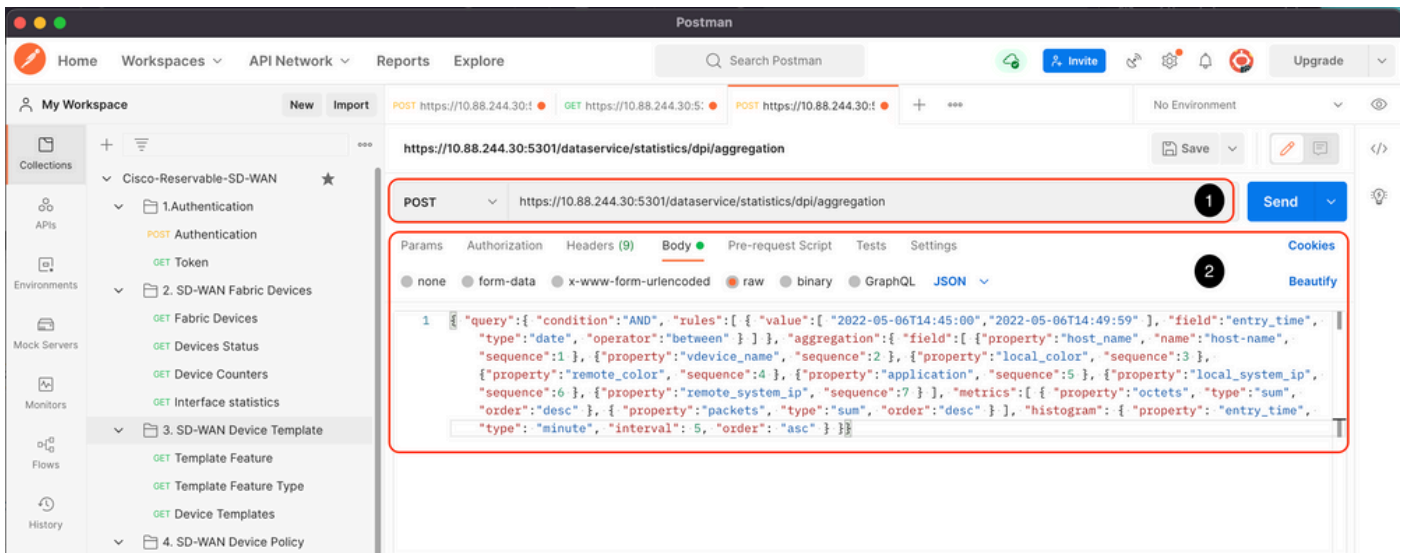
Cet exemple implique une requête POST

1. Sélectionnez l'appel API à exécuter, dans notre cas, <https://dataservice/statistics/dpi/aggregation>

**Conseil :** si vous souhaitez explorer d'autres appels API, accédez à l'URL vManage <https://vmanage-ip:port/apidocs>

2. Collectez le corps de votre appel API.

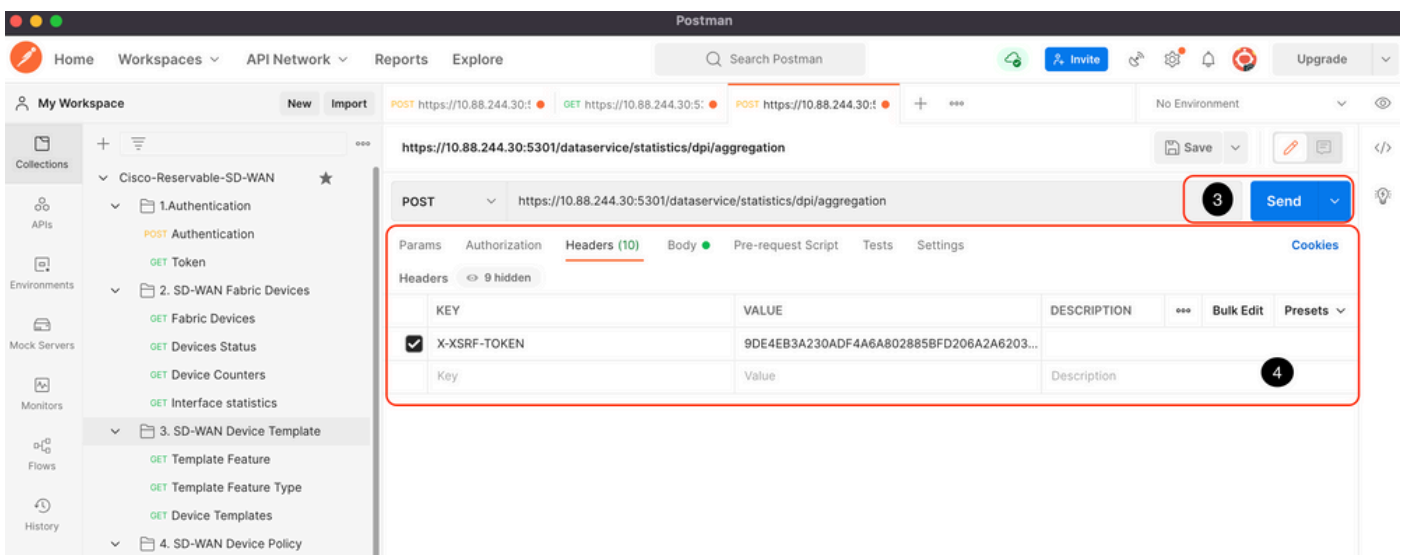
**Remarque :** cet appel API contient un corps au format JSON



3. Cliquez sur **Header** et ajoutez comme **Key** la chaîne X-XSRF-TOKEN en tant que valeur.

4. Cliquez sur **Envoyer**.

L'image affichée indique comment votre appel d'API doit apparaître.



Appel API d'agrégation DPI.

## Étape 5. Fermez votre session

Une fois que vous avez récupéré toutes les informations nécessaires à partir de vManage et/ou des périphériques, libérez des ressources de vManage et éliminez la possibilité pour des utilisateurs malveillants d'utiliser votre session.

## Exécuter des appels API dans un environnement automatisé

Enregistrer les cookies et les variables à utiliser dans les appels API suivants

### Comment enregistrer un jeton dans une variable ?

Enregistrez le jeton dans une variable pour une réutilisation ultérieure.

Enregistrer le jeton dans une variable

Lorsque nous demandons le jeton au format JSON, stockez-le. Utilisez l'onglet **Tests** et collez les lignes affichées.

```
var jsonData = JSON.parse(responseBody);
postman.setEnvironmentVariable("token", jsonData.token);
```

Ensuite, tout appel d'API peut utiliser une variable de jeton.

Key	Value	Description
<input checked="" type="checkbox"/> Host	<calculated when request is sent>	
<input checked="" type="checkbox"/> User-Agent	PostmanRuntime/7.26.3	
<input checked="" type="checkbox"/> Accept	*/*	
<input checked="" type="checkbox"/> Accept-Encoding	gzip, deflate, br	
<input checked="" type="checkbox"/> Connection	keep-alive	
<input checked="" type="checkbox"/> X-XSRF-TOKEN	{{token}}	
<input checked="" type="checkbox"/> Content-Type	application/json	

Utiliser la variable de jeton

## Comment effacer le cookie SESSIONID pour les nouvelles sessions ?

Chaque fois que vous exécutez l'appel API pour sortir de, utilisez JSESSIONID.

Nous ne pouvons pas utiliser d'authentification de base comme nous l'avons fait dans les versions précédentes. Au lieu de cela, nous fournissons uniquement des informations d'identification et enregistrons l'ID dans notre cookie. Avant cela, nous pouvons utiliser un pré-test pour effacer tous ou certains cookies.

The screenshot shows the Postman interface with a pre-request script editor. The script is as follows:

```

1  const jar = pm.cookies.jar();
2
3  jar.clear(pm.request.url, function (error) {
4    // error - <Error>
5  });

```

On the right side, there are links for "Pre-request scripts are written in JavaScript, and are run before the request is sent. Learn more about pre-request scripts" and a "SNIPPETS" section with options like "Get an environment variable", "Get a global variable", and "Get a variable".

Effacer les cookies

Cela se fait via le code placé dans le script de pré-demande.

## Comment utiliser Collection Runner ?

Maintenant que nous disposons d'un environnement dans lequel nous pouvons exécuter des sessions et enregistrer des données spécifiques à chaque session, vous pouvez exécuter une séquence d'appels à l'aide de Collection Runner.

Sélectionnez l'ordre des événements que vous voulez répéter, sélectionnez le nombre de répétitions afin que Postman puisse exécuter les appels API, le nombre sélectionné de fois avec des résultats par exécution.

The screenshot shows the Postman Collection Runner interface. On the left, there is a search bar and a list of collections under the "Viptela" environment. Below this, there are settings for "Environment" (SDWAN), "Iterations" (5), "Delay" (0 ms), and "Data" (Select File). There are also checkboxes for "Save responses", "Keep variable values" (checked), "Run collection without using stored cookies", and "Save cookies after collection run". A "Run Viptela" button is at the bottom.

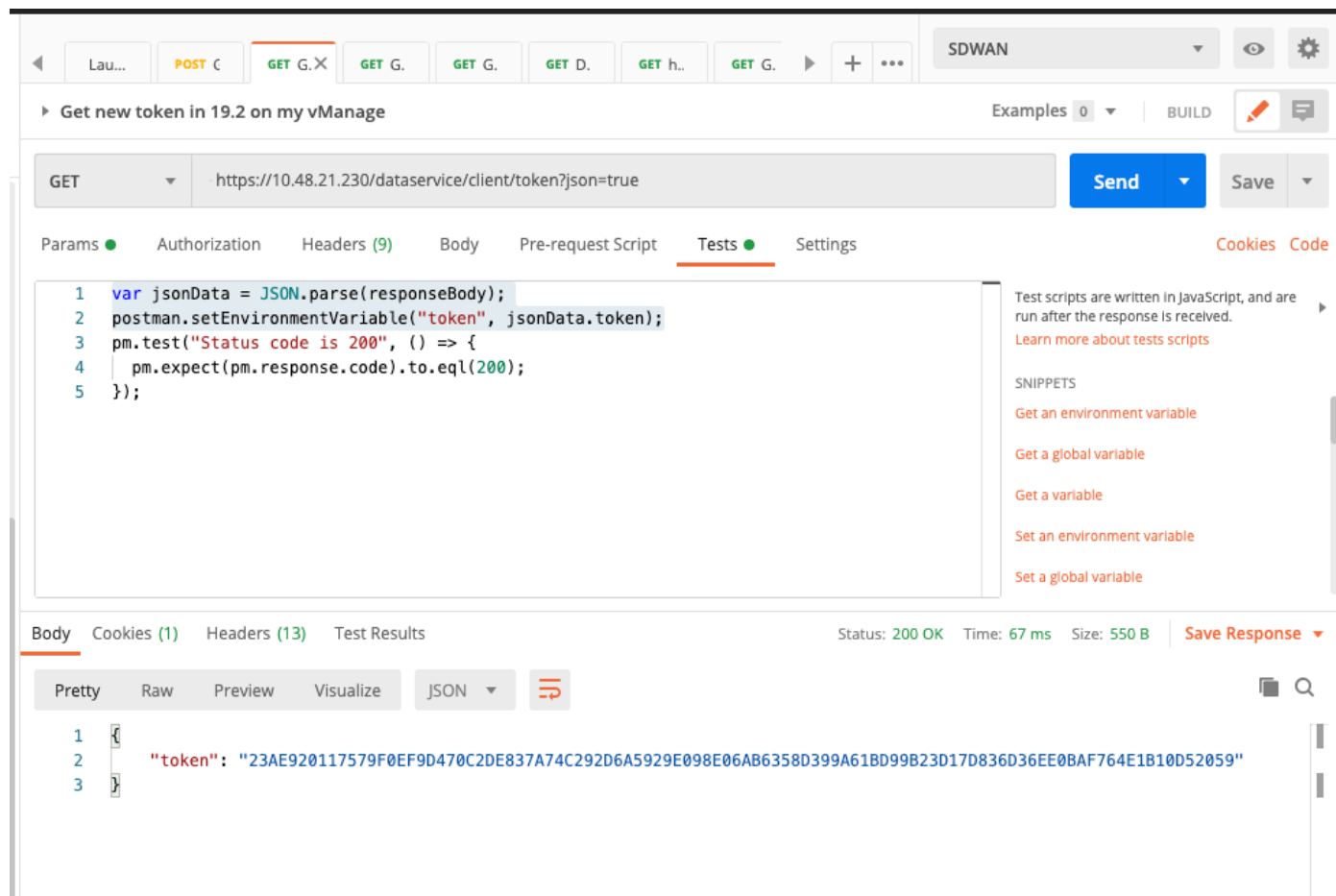
On the right, the "RUN ORDER" section shows a list of API calls with checkboxes to select their execution order. The first four items are selected:

- POST Get JSESSIONID in newer release(s)
- GET Get new token in 19.2 on my vManage
- GET Get server info with in-correct token
- GET Get server information 19.2 lab vManage with correct token

Other items in the list include various POST, PUT, and GET requests to different endpoints.

À partir de la « bibliothèque » d'appels, placez-les dans un certain ordre pour obtenir un flux/ordre spécifique à exécuter.

Indiquez si vous obtenez une valeur de 200 OK ou une autre valeur comme réponse et traitez-la comme réussite ou échec.



The screenshot shows the Postman interface for a REST client request. The request is a GET call to `https://10.48.21.230/dataservice/client/token?json=true`. The `Tests` tab is active, displaying a JavaScript test script:

```
1 var jsonData = JSON.parse(responseBody);
2 postman.setEnvironmentVariable("token", jsonData.token);
3 pm.test("Status code is 200", () => {
4   pm.expect(pm.response.code).to.eql(200);
5 });
```

The response status is `200 OK`, with a time of `67 ms` and a size of `550 B`. The response body is shown in JSON format:

```
1 {
2   "token": "23AE920117579F0EF9D470C2DE837A74C292D6A5929E098E06AB6358D399A61BD99B23D17D836D36EE0BAF764E1B10D52059"
3 }
```

Vérifier le code de réponse

```
pm.test("Status code is 200", () => {
  pm.expect(pm.response.code).to.eql(200);
});
```

Alors nous pouvons voir réussi ou échouer dans nos courses.



20 PASSED

0 FAILED

Viptela SDWAN  
just now

Run Summary

Export Results

Retry

New

Iteration 1

- POST Get JSESSIONID in newer release(s) https://10.48.21.230/j\_se... Viptela / Get JSESSIONID in newer ...
  - Status code is 200
- GET Get new token in 19.2 on my vManage https://10.48.21.230/dat... Viptela / Get new token in 19.2 on... 200 OK 53 ms 550 B
  - Status code is 200
- GET Get server info with in-correct token https://10.48.21.230/dat... Viptela / Get server info with in-co... 403 Forbidden 56 ms 583 B
  - Status code is 403
- GET Get server information 19.2 lab vManage with correct token https://10.48.21.230/dat... Viptela / Get server information 1... 200 OK 49 ms 486 B
  - Status code is 200

Iteration 2

- POST Get JSESSIONID in newer release(s) https://10.48.21.230/j\_se... Viptela / Get JSESSIONID in newer ...
  - Status code is 200
- GET Get new token in 19.2 on my vManage https://10.48.21.230/dat... Viptela / Get new token in 19.2 on... 200 OK 48 ms 550 B
  - Status code is 200
- GET Get server info with in-correct token https://10.48.21.230/dat... Viptela / Get server info with in-co... 403 Forbidden 49 ms 583 B
  - Status code is 403

Console

Exécution automatisée

À propos de cette traduction

Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.