Appels manuels de l'API XML vers CIMC

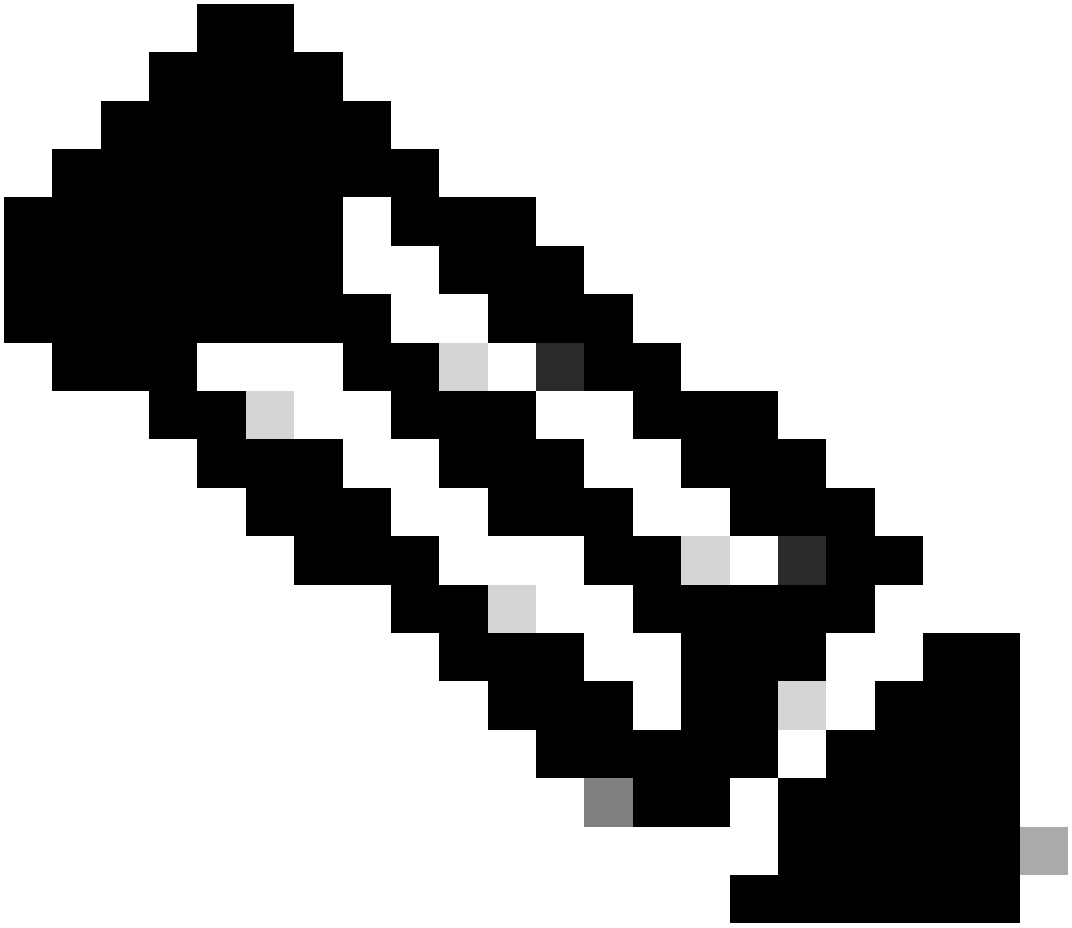
Table des matières

Introduction

Ce document décrit comment effectuer des appels manuels d'API XML vers le contrôleur de gestion intégré Cisco (CIMC).

Composants utilisés

- Une machine Linux (toute distribution).
- Connectivité réseau entre la machine Linux et Cisco CIMC (un test ping réussi suffit).



Remarque : Les noms de fichiers (main.sh, login.xml, get_summary.xml) utilisés dans ce guide sont arbitraires. Vous êtes libre d'utiliser vos propres conventions d'attribution de noms, à condition qu'elles soient correctement référencées tout au long du processus.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l’incidence possible des commandes.

Informations générales

Dans cette démonstration, un script Bash nommé "main.sh" est utilisé pour exécuter des appels API au serveur.

Veuillez noter que d'autres méthodes, telles que CURL (directement à partir de la CLI) ou Python, peuvent également être utilisées pour atteindre ce même résultat.

Appels d'API XML vers CIMC

L'API XML CIMC nécessite une étape d'authentification initiale à l'aide d'un appel d'API "aaaLogin". Au cours de ce processus, un cookie de session est récupéré, qui est ensuite utilisé pour authentifier tous les appels API suivants.

Étape 1

Commencez par créer un fichier nommé main.sh dans la machine Linux distante. Ce fichier contient un script bash utilisé pour envoyer les appels API.

Le script main.sh est exécuté plusieurs fois au cours de cet exercice : d'abord pour l'authentification, puis pour la récupération d'informations à partir du CIMC lors des appels suivants.

Voici le contenu du fichier main.sh :

```
#!/bin/bash
# CONFIGURATION - Edit these variables
CIMC_IP="X.X.X.X" # <<<<<<<< Customer CIMC IP address
USERNAME="admin"
PASSWORD="xxxxxxxx" # <<<<<<< CIMC Password in plaintext
XML_PAYLOAD_FILE="login.xml" # <<<<<<< Referencing the login.xml file for authentication and cookie ret
CIMC_URL="https://${CIMC_IP}/nuova" # <<<<<<< Call is made to this URL

# Check if payload file exists
if [ ! -f "$XML_PAYLOAD_FILE" ]; then
echo "Error: XML payload file '$XML_PAYLOAD_FILE' not found."
exit 1
fi
# Send XML request using curl
curl -k -s -u "${USERNAME}:${PASSWORD}" -H "Content-Type: text/xml" -d @"${XML_PAYLOAD_FILE}" "${CIMC_U
```

Le script bash définit les paramètres de connexion CIMC et exécute également un appel d'API XML, spécifié dans un autre fichier appelé login.xml identifié par une variable appelée XML_PAYLOAD_FILE.

Le script vérifie également si le fichier login.xml, défini par la variable XML_PAYLOAD_FILE, existe et est un fichier normal.

Si le fichier défini par XML_PAYLOAD_FILE n'existe pas, le script imprime une erreur et se ferme.

Enregistrez le fichier et rendez-le exécutable en exécutant la commande suivante dans l'interface de ligne de commande :

```
$ sudo chmod +x main.sh
```

Étape 2

Ensuite, créez le fichier appelé login.xml dans le même répertoire linux que le fichier main.sh.

Ce fichier de connexion contient l'appel d'API actualaaaLoginXML qui est envoyé au CIMC pour récupérer un cookie de session. Le cookie récupéré est utilisé pour les appels API suivants :

N’oubliez pas de remplacer le nom d’utilisateur et le mot de passe CIMC par les informations d’identification appropriées.

Exécutez le script main.sh dans CLI pour récupérer un cookie :

```
$ sudo ./main.sh
```

Si l'appel de l'API réussit, la réponse XML renvoyée contient une clé appelée outCookie, dont la valeur est le cookie récupéré comme indiqué :

```
<?xml version="1.0"?>
<aaaLogin cookie="" response="yes" outCookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" outRefreshPeriod="600" outPriv="admin" outSessionId="18"
outVersion="4.3(5.250001)"> </aaaLogin>
```

Dans le résultat, localisez la valeur outCookie.

Enregistrez cette valeur de cookie.

Étape 3

Créez un nouveau fichier dans le même répertoire que le fichier main.sh. Nommez le fichier get_summary.xml.

Pour obtenir une liste complète des demandes d'API pouvant être utilisées avec Cisco IMC, reportez-vous au Guide du programmeur d'API XML des serveurs rack Cisco UCS CIMC.

Le nouveau fichier get_summary.xml est utilisé pour récupérer les informations résumées du serveur et l'état de l'alimentation de l'hôte. en utilisant le bloc de code XML dans la documentation de référence, mais en remplaçant la valeur de clé de cookie par le cookie de récupération précédent.

Remplacez <cookie_value> par la valeur outCookie obtenue à partir du fichier précédent login.xmlresponse. La demande mise à jour ressemble à ceci :

```
<configResolveClass cookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
inHierarchical="false" classId="computeRackUnit"/>
```

Veillez à remplacer la <cookie_value> par votre valeur de cookie réelle récupérée lors du processus d'authentification.

Étape 4

Une fois que le cookie a été ajouté au nouveau fichier get_summary.xml, mettez à jour le script main.sh pour référencer le fichier "get_summary.xml" en tant que valeur pour la variable XML_PAYLOAD_FILE pour cette demande.

```
#!/bin/bash
# CONFIGURATION - Edit these variables
CIMC_IP="X.X.X.X"
USERNAME="admin"
PASSWORD="xxxxxxxx"
XML_PAYLOAD_FILE="get_summary.xml" # <<<<<<< Referencing the get_summary.xml file for subsequent API ca
CIMC_URL="https://${CIMC_IP}/nuova"
# Check if payload file exists
if [ ! -f "$XML_PAYLOAD_FILE" ]; then
echo "Error: XML payload file '$XML_PAYLOAD_FILE' not found."
exit 1
fi
# Send XML request using curl
curl -k -s -u "${USERNAME}:${PASSWORD}" -H "Content-Type: text/xml" -d @"${XML_PAYLOAD_FILE}" "${CIMC_U
```

Exécutez à nouveau le script main.sh pour exécuter la demande d'API mise à jour.

```
$ sudo ./main.sh
```

Vous recevez ensuite une réponse d'API au format XML renvoyant l'objet demandé par CIMC.

<?xml version="1.0"?>
<configResolveClass cookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" response="yes" classId="computeRackUnit">
<outConfigs>
<computeRackUnit dn="sys/rack-unit-1" adminPower="policy" availableMemory="524288" model="UCSC-C220-M6S" memorySpeed="3200" name="UCS C220 M6S" numOfAdaptors="2" numOfCores="56" numOfCoresEnabled="56" numOfCpus="2" numOfEthHostIfs="0" numOfFcHostIfs="0" numOfThreads="112" operPower="off" originalUuid="XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" presence="equipped" serverId="1" serial="XXXXXXXXXX" totalMemory="524288" usrLbl="" uuid="XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" vendor="Cisco Systems Inc" cimcResetReason="graceful-reboot " assetTag="Unknown" adaptorSecureUpdate="Enabled" resetComponents="components" storageResetStatus="NA" vicResetStatus="NA" bmcResetStatus="NA" smartUsbAccess="disabled" smartUsbStatus="Disabled" biosPostState="pending"/>
</outConfigs>
</configResolveClass>

Dépannage

Il est important de noter que l'appel de l'API XML aaaLogin renvoie un cookie de session avec un outRefreshPeriod d'environ 600 secondes. Cela signifie que le cookie expire au bout de dix (10) minutes s'il n'est pas actualisé, et qu'un nouveau cookie est requis pour continuer à exécuter des requêtes API.

Si vous tentez d'utiliser un cookie expiré (après 600 secondes), un bloc de réponse d'erreur XML 552 "Authorization required" est retourné :

<?xml version="1.0"?>
<configResolveDn cookie="xxxxxxxxx/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" response="yes" errorCode="552" invocationResult="service-unavailable" errorDescr="Authorization required"> </configResolveDn>

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.