Comment utiliser CISCO-BULK-FILE-MIB

Contenu

Introduction

Ce document explique comment utiliser CISCO-BULK-FILE-MIB et transférer les fichiers créés par cette base MIB (Management Information Base) à l'aide de CISCO-FTP-CLIENT-MIB.

Àpartir de la version 12.0 du logiciel Cisco IOS®, Cisco a mis en oeuvre un moyen de stocker un objet ou une table SNMP (Simple Network Management Protocol) sous forme de fichier sur le périphérique. Ce fichier peut ensuite être récupéré à l'aide de CISCO-FTP-CLIENT-MIB. Cette technologie vous permet de transférer de grandes quantités de données en utilisant une méthode de transport fiable.

Avant de commencer

Conditions requises

Avant d'essayer cette configuration, assurez-vous de respecter les conditions suivantes :

- Vous disposez d'un périphérique Cisco exécutant le logiciel Cisco IOS® version 12.0 ou ultérieure. Vérifiez l'outil de localisation MIB pour vous assurer que CISCO-BULK-FILE-MIB est pris en charge pour votre périphérique. Un lien vers l'outil se trouve sur la page Outils MIB de Cisco IOS.**Remarque :** cette MIB n'est pas prise en charge sur les périphériques Catalyst OS.

- SNMP doit être configuré sur le périphérique avec des chaînes de communauté en lecture seule et en lecture-écriture. Cette question n'est pas traitée dans ce document. Pour plus d'informations sur la configuration du protocole SNMP sur les périphériques IOS®, consultez Comment configurer les chaînes de communauté SNMP sur les routeurs, les commutateurs XL basés sur le logiciel Cisco IOS, les RSM, les MSFC et les commutateurs Catalyst.

Components Used

Les informations contenues dans ce document sont basées sur les versions de matériel et de logiciel suivantes :

- CISCO-BULK-FILE-MIB pour stocker le ifTable à partir d'un routeur 7507 exécutant 12.1(12) dans un fichier, puis utilisez le CISCO-FTP-CLIENT-MIB pour transférer ce fichier du routeur vers un serveur FTP.
- La suite de commandes SNMP net-snmp installée sous UNIX ou Windows.
- Ces MIB sont utilisées :SNMPv2-TCSNMPv2-SMISNMPv2-CONFSNMPv2-MIBIANAifType-MIBIF-MIBCISCO-SMICISCO-TCCISCO-BULK-FILE-MIBCISCO-FTP-CLIENT-MIB

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. If your network is live, make sure that you understand the potential impact of any command.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Informations générales

Assurez-vous que les MIB de ce tableau sont chargés dans votre plate-forme de gestion. Cela vous permet d'utiliser les noms et les valeurs d'objet listés ci-dessus au lieu des identificateurs d'objet numériques (OID). En général, ce document fait référence aux noms d'objets et non aux OID.

Format SMI de version 1	Format SMI version 2
SNMPv2-SMI-V1SMI.my	SNMPv2-SMI.my
SNMPv2-TC-V1SMI.my	SNMPv2-TC.my
	SNMPv2-CONF.my
SNMPv2-MIB-V1SMI.my	SNMPv2-MIB.my
IANAifType-MIB-V1SMI.my	IANAifType-MIB.my
IF-MIB-V1SMI.my	IF-MIB.my
CISCO-SMI-V1SMI.my	CISCO-SMI.my
CISCO-TC-V1SMI.my	CISCO-TC.my
CISCO-BULK-FILE-MIB-V1SMI.my	CISCO-BULK-FILE-MIB.my
CISCO-FTP-CLIENT-MIB-V1SMI.my	CISCO-FTP-CLIENT-MIB.my

Utilisation de CISCO-BULK-FILE-MIB

Création d'une opération de fichier en bloc

Dans cet exemple, nous capturons le `ifTable` à partir d'un routeur et le stockons dans un fichier en bloc. Cependant, vous pouvez utiliser n'importe quel objet ou table MIB.

Utilisez la version net-snmp de **snmpset**. L'adresse IP du routeur est **14.32.8.2**. Sa chaîne de communauté en lecture-écriture est **privée**. La chaîne de communauté en lecture seule est **publique**.

Chaque fois que vous créez une opération de fichier en bloc, choisissez deux nombres aléatoires pour l'instance de ligne. Il peut s'agir d'un nombre compris entre 1 et 4294967295 inclus. Pour les besoins de cet exemple, utilisez 333 et 444.

Step-by-Step Instructions

Pour créer une opération BULK-FILE, procédez comme suit :

1. Configurez le fichier à créer.
   ```
   $ snmpset -c private 14.32.8.2 cbfDefineFileEntryStatus.333 i 5
   $ snmpset -c private 14.32.8.2 cbfDefineFileName.333 s ifTable.txt
   $ snmpset -c private 14.32.8.2 cbfDefineFileFormat.333 i bulkASCII
   ```

2. Spécifiez l'objet MIB à capturer.Cet objet nécessite deux indices pour un fonctionnement correct. Le 333 est le 333 de la table de création de fichier ci-dessus. Le 444 est un nouveau nombre aléatoire utilisé pour l'index principal dans `cbfDefineObjectTable`.Cet exemple montre comment utiliser un nom d'objet pour `cbfDefineObjectID (ifTable)`. Vous pouvez également utiliser un OID complet ici.
   ```
   $ snmpset -c private 14.32.8.2 cbfDefineObjectID.333.444 o ifTable
   ```

3. Activez les lignes nouvellement créées.Vous devez avoir les deux indices pour votre ligne `cbfDefineObjectTable`.
   ```
   $ snmpset -c private 14.32.8.2 cbfDefineObjectEntryStatus.333.444 i 1
   $ snmpset -c private 14.32.8.2 cbfDefineFileEntryStatus.333 i 1
   ```

4. Créez le fichier.
   ```
   $ snmpset -c private 14.32.8.2 cbfDefineFileNow.333 i 3
   ```

   Le fichier en bloc est créé.
5. Vérifiez que le fichier a bien été créé à l'aide de **snmpget** sur l'objet `cbfStatusFileState`.Cet objet nécessite deux indices. Le premier index est le nombre aléatoire choisi pour la table Fichier (333 dans cet exemple). Le deuxième index dépend du nombre de fichiers que vous avez créés sur votre routeur. Comme il s'agit de votre premier fichier, l'index est 1. Par conséquent, utilisez la commande suivante :
   ```
   $ snmpget -c public 14.32.8.2 cbfStatusFileState.333.1
   ```

   La valeur `running(1)` signifie que le fichier est en cours de création. Une valeur de `ready(2)` signifie que le fichier a été créé avec succès et qu'il est en attente de lecture.

Ce fichier n'est toutefois pas directement accessible depuis le routeur. Utilisez CISCO-FTP-

CLIENT-MIB pour lire ce fichier.

Transfert du fichier à l'aide de CISCO-FTP-CLIENT-MIB

Pour chaque opération du client FTP, vous devez sélectionner un nombre aléatoire pour l'instance de ligne. Vous pouvez utiliser l'un des mêmes nombres aléatoires que ceux que vous avez utilisés ci-dessus. Cet exemple utilise 555.

Step-by-Step Instructions

Pour transférer le fichier à l'aide d'une base de données CISCO-FTP-CLIENT-MIB, procédez comme suit :

1. Créez une instance de ligne du client FTP.
   ```
   $ snmpset -c private 14.32.8.2 cfcRequestEntryStatus.555 i 5
   ```

2. Complétez les paramètres requis. Le fichier local **doit** être du même nom que le fichier que vous avez créé ci-dessus ! Utilisez **putASCII** pour transférer des fichiers `ASCII en masse`.Si vous définissez `cbfDefineFileFormat` sur `vracBinary` ci-dessus, vous devez définir `cfcRequestOperation` sur `putBinary`.
   ```
   $ snmpset -c private 14.32.8.2 cfcRequestOperation.555 i putASCII
   $ snmpset -c private 14.32.8.2 cfcRequestLocalFile.555 s ifTable.txt
   $ snmpset -c private 14.32.8.2 cfcRequestRemoteFile.555 s /home/Marcus/ifTable.txt
   $ snmpset -c private 14.32.8.2 cfcRequestServer.555 s 172.18.123.33
   $ snmpset -c private 14.32.8.2 cfcRequestUser.555 s Marcus
   $ snmpset -c private 14.32.8.2 cfcRequestPassword.555 s marcus123
   ```

3. Commencez le transfert en définissant la ligne sur active.
   ```
   $ snmpset -c private 14.32.8.2 cfcRequestEntryStatus.555 i 1
   ```

   Le transfert FTP commence. Une fois terminé, le fichier est enregistré sur **/home/Marcus/ifTable.txt**.
4. Pour obtenir l'état du transfert FTP, utilisez **snmpget** à nouveau sur l'objet `cfcRequestResult`.Cet objet utilise le même index que celui utilisé avec les autres objets FTP.
   ```
   $ snmpget -c public 14.32.8.2 cfcRequestResult.555
   ```

   Une valeur `en attente(1)` signifie que le fichier est toujours en cours de transfert. Une valeur de `réussite(2)` signifie que le fichier a été transféré avec succès. Toute autre valeur est une erreur.
5. Une fois le transfert terminé, essayez à nouveau **snmpget** de l'objet `cbfStatusFileState`. Il a maintenant une valeur différente.
   ```
   $ snmpget -c public 14.32.8.2 cbfStatusFileState.333.1
   enterprises.cisco.ciscoMgmt.ciscoBulkFileMIB.ciscoBulkFileMIBObjects.cbfStatus.
   cbfStatusFileTable.cbfStatusFileEntry.cbfStatusFileState.333.1 = emptied(3)
   ```
   La valeur de `vidé(3)` signifie que le fichier a été lu correctement. Le fichier ne peut pas être transféré à nouveau.
6. Il est désormais possible de supprimer ce fichier en détruisant la ligne d'état du fichier. Cet objet prend les mêmes indices que le `cbfStatusFileState` ci-dessus.
   ```
   $ snmpset -c private 14.32.8.2 cbfStatusFileEntryStatus.333.1 i 6
   ```

7. Une fois le fichier supprimé, supprimez les lignes Objet et Fichier correspondantes.
   ```
   $ snmpset -c private 14.32.8.2 cbfDefineObjectEntryStatus.333.444 i 6
   ```

```
$ snmpset -c private 14.32.8.2 cbfDefineFileEntryStatus.333 i 6
```

De cette manière, vous pouvez utiliser CISCO-FTP-CLIENT-MIB pour transférer n'importe quel fichier hors du routeur via FTP.

Vérification du résultat

Cette section vous guide tout au long de la lecture de la syntaxe de ce fichier.

1. La première ligne est la ligne `de préfixe`. Pour notre exemple `ifTable`, il s'agit :
   `prefix 1.3.6.1.2.1.2.2.1`
   Cela correspond à l'OID de l'objet `ifEntry`. Le `ifTable` est composé d'une ou plusieurs `ifEntries`.
2. La ligne suivante répertorie le nombre d'objets dans le tableau. La ligne se compose de la table de mots-clés suivie du nombre d'objets dans la table, suivi de l'index de chaque objet.Exemple :
   `table 22 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22`
   Cette ligne indique que la table contient 22 objets et que chaque objet a un index incrémentant. Ces objets proviennent de l'exemple `ifTable` :
   ```
   ifIndex
   ifDescr
   ifType
   ifSpeed
   ...
   ```
3. Après cette ligne, il y a plusieurs entrées de ligne. Dans l'exemple `ifTable`, chaque ligne correspond à une interface. Les lignes commencent par la ligne de mot clé, suivie de leur identificateur d'index, puis des objets énumérés par l'entrée de table précédente.Exemple :
   `row 1 1 546F6B656E52696E67302F30 9 4464 16000000 0008B0851800 2 2 6551 0 0 0 0 0 0 0 0 0 0 0 0.0`
4. La quatrième entrée est le `ifDescr` pour l'interface 1. Cependant, il s'agit du `ifDescr` en ASCII codé au format hexadécimal.Pour traduire cette ligne dans un format plus lisible, utilisez la commande Perl suivante :
   ```
   $ perl -e 'print pack("H*", "546F6B656E52696E67302F30")'
   TokenRing0/0
   ```
   Cette entrée correspond à l'interface TokenRing0/0. Tous les objets qui sont normalement des chaînes sont affichés sous forme d'ASCII codé en hexadécimal dans les fichiers en bloc. Vous pouvez utiliser cette commande Perl pour traduire n'importe quelle chaîne ASCII hexadécimale en texte lisible. Si vous n'avez pas Perl, utilisez cette table de caractères ASCII pour traduire la chaîne.
5. Certaines entrées affichent ~ caractères pour les valeurs. Cela signifie que la valeur de cet objet est NULL. Autrement dit, l'objet n'est pas instancié sur le périphérique.Exemple :
   `row 9 9 41544D312F302F302D61746D206C61796572 37 ~ 0  1 1 5971 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~`
   Cela correspond à l'interface de couche ATM1/0/0-atm. Notez que `ifMtu` est NULL pour cette interface. Comme il s'agit d'une interface virtuelle, il est logique qu'elle ne possède pas de MTU. Si vous préférez, vous pouvez remplacer ces valeurs NULL par **0** en ajoutant cette commande à la configuration du périphérique :
   ```
   Router(config)#no snmp-server sparse-table
   ```

Dépannage du résultat

Lors de l'interrogation de l'objet `cbfStatusFileState`, si vous recevez une valeur autre que `running(1)`, `ready(2)` ou `vidé(3)`, votre opération a rencontré une erreur. Voici les causes des erreurs :

```
noSpace    no data due to insufficient file space
badName    no data due to a name or path problem
writeErr   no data due to fatal file write error
noMem      no data due to insufficient dynamic memory
buffErr    implementation buffer too small
aborted    short terminated by operator command
```

Si le nombre d'objets dans le fichier est inférieur à ce que vous attendez, `cbfDefineMaxObjects` de CISCO-BULK-FILE-MIB peut être défini sur un nombre trop faible. Pour déterminer la valeur actuelle de l'objet, utilisez **snmpget**.

```
$ snmpget -c public 14.32.8.2 cbfDefineMaxObjects.0
```

La valeur 0 signifie qu'aucune limite n'est configurée. La valeur peut être définie sur n'importe quel entier compris entre 0 et 4294967295. Pour définir le nombre maximal d'objets par fichier sur 10, utilisez la commande **snmpset**. L'index de cet objet est toujours 0.

```
$ snmpset -c private 14.32.8.2 cbfDefineMaxObjects.0 u 10
```

Cet objet peut ne pas être configurable sur toutes les plates-formes. Si **snmpset** échoue avec cette erreur, l'objet n'est pas configurable sur votre plate-forme :

```
Error in packet.
Reason: (noSuchName) There is no such variable name in this MIB.
Failed object:
enterprises.cisco.ciscoMgmt.ciscoBulkFileMIB.ciscoBulkFileMIBObjects.cbfDefine.cbfDefineMaxObjec
ts.0
```

Lors de l'interrogation de l'objet `cfcRequestResult`, si vous recevez une valeur autre que `en attente(1)` ou `réussie(2)`, l'opération FTP a rencontré une erreur. Voici les causes des erreurs :

```
aborted              user aborted the transfer
fileOpenFailLocal    local bulk file was not found
fileOpenFailRemote   remote file could not be opened for writing
badDomainName        FTP server's hostname could not be resolved
unreachableIpAddress route to the FTP server could not be found
linkFailed           connection could not be made to the remote server
fileReadFailed       local file could not be read
fileWriteFailed      remote file could not be written
```

Cavates

- Il n'existe actuellement aucun moyen d'accéder directement aux fichiers en bloc. Vous devez passer par CISCO-FTP-CLIENT-MIB pour lire les fichiers.
- L'objet `cbfDefineFileStorage` définit trois types : `éphémère`, `volatile` et `permanente`. Actuellement, le seul type pris en charge dans IOS est `éphémère`. Les fichiers éphémères existent en petites

quantités jusqu'à ce qu'ils soient lus.
- Une fois les fichiers lus, ils ne peuvent pas être relus. Il faut d'abord les recréer.
- L'objet `cbfDefineFileFormat` définit trois types : `standardBER`, `vracBinary` et `vracASCII`. Les seuls formats pris en charge sont `vracBinary` et `vracASCII`. Le format par défaut est `vracBinary`.
- Le serveur FTP Chameleon pour Windows est connu **ne** fonctionne **pas** avec CISCO-FTP-CLIENT-MIB, car il ne retourne pas les codes de résultat corrects.

Informations connexes

- Comment configurer des chaînes de communauté SNMP sur des routeurs, des commutateurs XL basés sur le logiciel Cisco IOS, des RSM, des MSFC et des commutateurs Catalyst
- Support technique - Cisco Systems