SNMP を使用してハングしている TCP 接続を検出しクリアする方法

目次

はじめに

このドキュメントでは、Cisco IOS デバイスでハングした TCP 接続を検出してクリアするために Simple Network Management Protocol（SNMP; 簡易ネットワーク管理プロトコル）を使用する方法について説明します。 また、この目的のために使用する SNMP オブジェクトについても説明します。

「ハングした TCP 接続を検出してクリアする PERL スクリプト」というタイトルのセクションでは、これらの手順を実装している PERL スクリプトへのリンクを提供しています。

前提条件

要件

このドキュメントの読者は次のトピックについて理解している必要があります。

- Cisco デバイス上で TCP 接続情報を表示する方法の理解
- SNMP の **walk**、**get**、**get-next**、**set** コマンドの一般的な使用方法
- Cisco デバイス上での SNMP の設定方法の理解

使用するコンポーネント

このドキュメントは、TCP-MIB モジュールと CISCO-TCP-MIB モジュールをサポートしている IOS ソフトウェアを稼働している Cisco ルータとスイッチに適用されます。

注: CISCO-TCP-MIB モジュールは、デフォルトで NET-SNMP にはロードされません。 MIB モジュールがシステムにロードされない場合、オブジェクトの名前ではなく OID を使用してオブジェクトを参照する必要があります。

このドキュメントの情報は、すべての IOS 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。

情報は、NET-SNMP の次のバージョンに基づいています。

- http://www.net-snmp.org/ で利用可能な NET-SNMP バージョン 5.1.2

PERL スクリプトは次の PERL バージョンでテストされました。

- FreeBSD での 5.005_03
- Solaris 5.8 での 5.8.0
- 5.005_02：Microsoft Windows 2000 上で CiscoWorks SNMS の一部として出荷
- http://www.activestate.com/Products/ActivePerl/ で利用可能な Microsoft Windows 2000 の ActivePerl 5.8.4 。

本書の情報は、特定のラボ環境にあるデバイスに基づいて作成されたものです。 このドキュメントで使用するすべてのデバイスは、初期（デフォルト）設定の状態から起動しています。 稼働中のネットワークで作業を行う場合、コマンドの影響について十分に理解したうえで作業してください。

表記法

ドキュメント表記の詳細は、『シスコ テクニカル ティップスの表記法』を参照してください。

背景説明

MIB オブジェクトの詳細：オブジェクト識別子（OID）を含む

使用するオブジェクトは次のとおりです。

CISCO-TCP-MIB モジュールからの場合：

- ciscoTcpConnInBytes、OID .1.3.6.1.4.1.9.9.6.1.1.1.1この接続への入力バイト数。
- ciscoTcpConnInPkts、OID 1.3.6.1.4.1.9.9.6.1.1.1.2この接続への入力パケット数。
- ciscoTcpConnOutBytes、OID .1.3.6.1.4.1.9.9.6.1.1.1.3この接続からの出力バイト数。
- ciscoTcpConnOutPkts、OID .1.3.6.1.4.1.9.9.6.1.1.1.4この接続からの出力パケット数。
- ciscoTcpConnRetransPkts、OID .1.3.6.1.4.1.9.9.6.1.1.1.7この接続で再転送されるパケット数。
- ciscoTcpConnRto、OID .1.3.6.1.4.1.9.9.6.1.1.1.9この接続用の再転送タイムアウト値。

TCP-MIB モジュールからの場合：

- tcpConnState、OID .1.3.6.1.2.1.6.13.1.1この接続に対するステータス。

これらのオブジェクトの詳細については、「MIB オブジェクトの詳細情報」を参照してください。

TCP 接続のハングを検出するために SNMP を使用する

要約

次のステップを使用すると、TCP 接続がハングするかどうかを判断できます。

1. ciscoTcpConnRetransPkts オブジェクトと ciscoTcpConnRto オブジェクトがデバイスでサポートされているかどうかを確認するために、**ciscoTcpConnRto** に対して SNMP get-next 動作を実行して、何らかのオブジェクトが戻されることを確認します。**注:** どちらのオブジェクトも同時に追加されたため、確認が必要なのは 1 つのオブジェクトだけです。**注:** すべての Cisco デバイスが最後の 2 つのオブジェクト（ciscoTcpConnRetransPkts および ciscoTcpConnRto）をサポートするわけではありませんが、それらを使用することで、検出の正確性が高まります。ciscoTcpConnRetransPkts オブジェクトと ciscoTcpConnRto オブジェクトがサポート**されている**場合、ステップ 2 に進みます。ciscoTcpConnRetransPkts オブジェクトと ciscoTcpConnRto オブジェクトがサポート**されていない**場合、ステップ 3 に進みます。
2. すべてのオブジェクトがサポートされています。 それぞれの TCP 接続に対して次を確認します。ciscoTcpConnOutBytes が 0 である。ciscoTcpConnOutPkts が 0 である。ciscoTcpConnRetransPkts が 0 よりも大きい。ciscoTcpConnRto が 20,000 よりも大きい。**注:** 検出を高速化するのであれば、20,000 を減らすことができます。 接続がハングすると、Rto が 20,000 に達するまで 1 分間ほどかかります。 ただし、小さな値によって、結果の正確さが低くなる可能性があります。前述のすべてが正しい場合、この TCP 接続はハングしており、クリアできます。 「ハングしている TCP 接続をクリアするために SNMP を使用する」に進んでください。
3. 最初の 4 つのオブジェクトだけがサポートされています。 それぞれの TCP 接続に対して次を確認します。ciscoTcpConnInBytes が 0 よりも大きい。ciscoTcpConnInPkts が 0 である。ciscoTcpConnOutBytes が 0 である。ciscoTcpConnOutPkts が 0 である。数秒待ってからもう一度オブジェクトを **get** して、それが確立されるプロセスでの TCP 接続ではなかったことを確認します。**注:** 最初の 2 つのチェック（入力バイトであり、入力パケットではない正の数値）は異常に見えるかもしれませんが、さまざまなデバイスと IOS バージョンに対して確認されたものです。**注:** 6 つのオブジェクトすべてをサポートする IOS バージョンは、この動作を示さないため、ステップ 2 のテストはこれらの最初の 2 つのテストを含みません。どちらの場合もすべてのオブジェクトがテストを満たす場合、この TCP 接続はハングしており、クリアできます。 「ハングしている TCP 接続をクリアするために SNMP を使用する」に進んでください。

手順説明

この例の値は次のとおりです。

- デバイス ホスト名 a = nms-7206a（すべてのプロジェクトをサポート）
- デバイス ホスト名 b = nms-1605（最初の 4 つのオブジェクトだけをサポート）
- 読み取りコミュニティ = public
- 書き込みコミュニティ = private

コミュニティ ストリングとホスト名を次のコマンドで置き換えます。

1. このデバイスが ciscoTcpConnRetransPkts オブジェクトと ciscoTcpConnRto オブジェクトをサポートするかどうかを判断します。**ciscoTcpConnRto** に対して SNMP get-next 動作を実行します。

   ```
   snmpgetnext -c public nms-7206a ciscoTcpConnRto
   ```

   オブジェクトがサポート**されている**場合、次のような応答になります。

   ```
   snmpgetnext -c public nms-7206a ciscoTcpConnRto
   ```

   注: これらのオブジェクトに使用されるインデックスは、この場合は 14.32.100.75.2065.172.18.86.111.23092 ですが、ローカル IP アドレス（14.32.100.75）、ローカル TCP ポート番号（2065）、リモート IP アドレス（172.18.86.111）、およびリモートの TCP ポート番号（23092）を結合させたものです。戻り値は ciscoTcpConnRto 用です。 ステップ 2 に進みます。オブジェクトがサポート**されていない**場合、次のような応答になります。

   ```
   snmpgetnext -c public nms-1605 ciscoTcpConnRto
   CISCO-FLASH-MIB::ciscoFlashDevicesSupported.0 = INTEGER: 1
   ```

   戻り値は **ciscoTcpConnRto** オブジェクト用ではありません。 戻される正確なオブジェクトは重要ではありません。 ステップ 3.に進んで下さい。
2. Cisco TCP 接続テーブルで 6 つのオブジェクトすべてをサポートするデバイスに対する各 TCP 接続に関する情報を **get** します。**ciscoTcpConnOutBytes**、ciscoTcpConnOutPkts、ciscoTcpConnRetransPkts、および ciscoTcpConnRto に対して SNMP get-next 動作を実行します。

   ```
   snmpgetnext -c public nms-7206a ciscoTcpConnOutBytes
                        ciscoTcpConnOutPkts
                        ciscoTcpConnRetransPkts
                        ciscoTcpConnRto
   ```

   次のような応答になります。

   ```
   snmpgetnext -c public nms-7206a ciscoTcpConnOutBytes
                        ciscoTcpConnOutPkts
                        ciscoTcpConnRetransPkts
                        ciscoTcpConnRto
   ```

   次の内容を確認します。ciscoTcpConnOutBytes が 0 である。ciscoTcpConnOutPkts が 0 である。ciscoTcpConnRetransPkts が 0 よりも大きい。ciscoTcpConnRto が 20,000 よりも大きい。**注:** 検出を高速化するのであれば、20,000 を減らすことができます。 接続がハングすると、Rto が 20,000 に達するまで 1 分間ほどかかります。 ただし、小さな値によって、結果の正確さが低くなる可能性があります。これらのすべてが正しい場合、この TCP 接続はハングしており、クリアできます。 「ハングしている TCP 接続をクリアするために SNMP を使用する」に進んでください。TCP 接続テーブルの **walk** を続行します。 これを実行するために、ハングした接続を確認するときに、次のような戻されたオブジェクトを使用して繰り返し SNMP **get-next** 動作を実行します。

```
snmpgetnext -c public nms-7206a ciscoTcpConnOutBytes.14.32.100.75.2065.172.18.86.111.23092
                    ciscoTcpConnOutPkts.14.32.100.75.2065.172.18.86.111.23092
                    ciscoTcpConnRetransPkts.14.32.100.75.2065.172.18.86.111.23092
                    ciscoTcpConnRto.14.32.100.75.2065.172.18.86.111.23092
```

get-next 動作がこの方式でオブジェクトを返すまで、これまでのテストを使用して各エントリを確認します。

```
snmpgetnext -c public nms-7206a ciscoTcpConnOutBytes.14.32.100.75.2065.172.18.86.111.23092
                    ciscoTcpConnOutPkts.14.32.100.75.2065.172.18.86.111.23092
                    ciscoTcpConnRetransPkts.14.32.100.75.2065.172.18.86.111.23092
                    ciscoTcpConnRto.14.32.100.75.2065.172.18.86.111.23092
```

これで、このデバイスの TCP 接続をすべて確認して終了しました。

3. Cisco TCP 接続テーブルで最初の 4 つのオブジェクトだけをサポートするデバイスに対する各 TCP 接続に関する情報を **get** します。**ciscoTcpConnInBytes**、ciscoTcpConnInPkts、ciscoTcpConnOutBytes、および ciscoTcpConnOutPkts に対して SNMP get-next 動作を実行します。

```
snmpgetnext -c public nms-1605 ciscoTcpConnInBytes
                    ciscoTcpConnInPkts
                    ciscoTcpConnOutBytes
                    ciscoTcpConnOutPkts
```

次のような応答になります。

```
snmpgetnext -c public nms-1605 ciscoTcpConnInBytes
                    ciscoTcpConnInPkts
                    ciscoTcpConnOutBytes
                    ciscoTcpConnOutPkts
```

次の内容が正しいことを確認します。ciscoTcpConnInBytes が 0 よりも大きい。ciscoTcpConnInPkts が 0 である。ciscoTcpConnOutBytes が 0 である。ciscoTcpConnOutPkts が 0 である。数秒待って、もう一度オブジェクトを **get** します。 確立される処理でそれが TCP 接続ではなかったことを確認します。前述のすべてが**正しい**場合、この TCP 接続はハングしており、クリアできます。 「ハングしている TCP 接続をクリアするために SNMP を使用する」に進んでください。TCP 接続テーブルの **walk** を続行します。 これを実行するために、ハングした接続を確認するときに、次のような戻されたオブジェクトを使用して繰り返し SNMP **get-next** 動作を実行します。

```
snmpgetnext -c public nms-1605 ciscoTcpConnInBytes.14.32.6.185.23.14.32.100.33.2249
                    ciscoTcpConnInPkts.14.32.6.185.23.14.32.100.33.2249
                    ciscoTcpConnOutBytes.14.32.6.185.23.14.32.100.33.2249
                    ciscoTcpConnOutPkts.14.32.6.185.23.14.32.100.33.2249
```

get-next 動作がこの方式でオブジェクトを返すまで、これまでのテストを使用して各エントリを確認します。

```
snmpgetnext -c public nms-1605 ciscoTcpConnInBytes.14.32.6.185.23.14.32.100.33.2249
                    ciscoTcpConnInPkts.14.32.6.185.23.14.32.100.33.2249
                    ciscoTcpConnOutBytes.14.32.6.185.23.14.32.100.33.2249
                    ciscoTcpConnOutPkts.14.32.6.185.23.14.32.100.33.2249
```

これで、このデバイスの TCP 接続をすべて確認して終了しました。

ハングしている TCP 接続をクリアするために SNMP を使用する

手順説明

SNMP を使用すると、ハングした TCP 接続をクリアできます。 SNMP コマンドは、**clear tcp local <local_ip> <local_port> remote <remote_ip> <remote_port>** コマンドと同じです。 1 行をクリアするために使用するオブジェクトは **tcpConnState** です。

SNMP でハングした TCP 接続をクリアするには、次のコマンドを発行します。

```
snmpset -c private nms-7206a tcpConnState.14.32.100.75.2065.172.18.86.111.23092 integer
deleteTCB
TCP-MIB::tcpConnState.14.32.100.75.2065.172.18.86.111.23092 = INTEGER: deleteTCB(12)
```

注: これらのオブジェクトに使用されるインデックスは、この場合は 14.32.100.75.2065.172.18.86.111.23092 ですが、ローカル IP アドレス（14.32.100.75）、ローカル TCP ポート番号（2065）、リモート IP アドレス（172.18.86.111）、およびリモートの TCP ポート番号（23092）を結合させたものです。

注: 「TCP 接続がハングするかを検出するために SNMP を使用する」でハングしたと判断したインデックスを正確に使用する必要があります。 このコマンドが警告なく TCP 接続を切断することに注意してください。

MIB オブジェクトの詳細情報


```
snmpset -c private nms-7206a tcpConnState.14.32.100.75.2065.172.18.86.111.23092 integer
deleteTCB
TCP-MIB::tcpConnState.14.32.100.75.2065.172.18.86.111.23092 = INTEGER: deleteTCB(12)
```


ハングした TCP 接続を検出してクリアする PERL スクリプト

このリンクは、PERL スクリプトと必要な MIB モジュールのあるアーカイブ ファイルを提供します。 リンクを右クリックしてファイルをシステムに保存してください。

- fixTCPhang.tgz

アーカイブに含まれるファイルは次のとおりです。

- bin/fixTCPhang.pl
- mibs/CISCO-SMI.my
- mibs/CISCO-TCP-MIB.my

スクリプトと MIB モジュールを抽出するには、UNIX 系のオペレーティング システムでは gzip や tar のようなユーティリティを使用します。 たとえば、アーカイブ ファイルが **/tmp** にあると仮定してファイルを **/tmp** に抽出するには、次のように行います。

```
cd /tmp; gzip -dc fixTCPhang.tgz | tar -xvf -
```

注: PERL の場所を指定するために、スクリプトの最初の行を編集する必要があります。

Microsoft Windows オペレーティング システムでは、ファイルの抽出に winzip などのユーティリティを使用します。 c:\tmp にファイルを抽出した場合は、スクリプトを実行するときに -m オプションを指定する必要がありません。

次のコマンドでファイルを起動します。

```
fixTCPhang.pl -c public -C private -f nms-7206a
```

検出したハングした TCP 接続それぞれに対して、次の出力のような行が表示されます。

```
fixTCPhang.pl -c public -C private -f nms-7206a
```

読み書きのコミュニティ ストリングが提供されて、-f オプションが指定された場合、スクリプトは接続をクリアしています。 出力の最後にある CLEARED 文に注目してください。

スクリプトは SNMP バージョン 1、2c、3 をサポートしています。 SNMP バージョン 3 を指定する場合、-v 引数に他のすべての認証情報を指定する必要があります。 次は SNMP v3 を使用する例です。

```
fixTCPhang.pl -v "3 -a MD5 -u chelliot -A chelliot -l authNoPriv" -f nms-dmz-ap1200-b
```

前述の例で SNMP v3 を設定するための IOS コマンドは次のとおりです。

```
snmp-server group chelliot-group v3 auth write v1default
snmp-server user chelliot chelliot-group v3 auth md5 chelliot
```

注: このテストで使用される NET-SNMP の Windows バージョンには不具合があることが示されています。 このバグによって、SHA 認証は正しく動作できません。

このスクリプトで使用できるオプションは他にいくつかあります。 NET-SNMP コマンドライン ユーティリティの場所や MIB モジュールの場所が /tmp/mibs にない場合に含めるスクリプト オプションがいくつかあります。 また、それらのオプションの次のような要約も表示できます。

```
fixTCPhang.pl
fixTCPhang.pl [-dfhV -c <read_community> -C <write_community> -m <mib_directory>
               -p <command_path> -t <timeout> -v <snmp_version>] <device>
```

```
Version 1.2
Detect hung TCP connections on <device>, optionally clearing them.
Options:  -c  Specify read community string. Defaults to public.
          -C  Specify the readwrite community string. No default.
               Must be supplied for the script to clear hung connections.
          -d  Turn on debug mode.
          -f  Fix or clear any hung TCP connections found.
          -h  Print this message.
          -m  Specify the directory to find CISCO-SMI.my and CISCO-TCP-MIB.my.
               Defaults to /tmp/mibs.
          -p  Where to find the net-snmp utilities.
               Optional if the utilities are in the path.
          -t  SNMP Timeout value. Defaults to 5 sec.
          -v  Specify SNMP version to use: One of 1, 2c, or 3.
               If 3 is specified then this option must include all of the
               authentication information for SNMPv3. For example:
               "3 -a MD5 -u chelliot -A chelliot -l authNoPriv"
               Note: NET-SNMP seems to have a bug with SHA authentication on Windows.
               See the NET-SNMP documentation for more information.
               Defaults to SNMP version 1.
          -V  Print version number.
```

関連情報

- テクニカルサポート - Cisco Systems