ベースライン プロセスのベスト プラクティスのホワイト ペーパー

目次

概要

この文書では、可用性の高いネットワークを構築するためのベースラインのコンセプトと手順を説明します。 これには、成功の評価を行うためのネットワーク ベースラインとしきい値設定に関する重要な成功要因が含まれています。 さらに、シスコのハイ アベイラビリティ サービス（HAS）チームによって明らかにされた最適な方法のガイドラインに基づき、ベースラインとしきい値プロセスおよび実装について詳細に説明します。

この文書では、ベースラインのプロセスを手順を追って実行します。 現在の Network Management System（NMS; ネットワーク管理システム）の製品によっては、このプロセスを自動化できますが、自動ツール、手動ツールのいずれを使用しても、ベースラインのプロセス自体は同じです。 これらの NMS 製品を使用する場合は、それぞれの独自のネットワーク環境に合わせてデフォルトのしきい値設定を調整する必要があります。 しきい値が有意義で正しくなるよう

に、しきい値をインテリジェントに選択するプロセスを導入することが重要です。

基準

ベースラインとは何か

ベースラインとは、ネットワークを定期的に調査してネットワークが確実に設計どおりに動作するようにするためのプロセスです。 ベースラインは、特定時点のネットワークの状態を詳細に言及する一つのレポートにとどまりません。 ベースライン プロセスに従って入手できる情報は次のとおりです。

- ハードウェアとソフトウェアの正常性に関する有用な情報を取得する。
- 現在のネットワーク リソースの使用状況を判別する。
- ネットワーク アラームしきい値に関して正確な決定を下す。
- 現在発生しているネットワークの問題を特定する。
- 将来の問題を予測できる。

ベースラインのもう 1 つのとらえ方を次の図に示します。

ネットワークのブレーク ポイントを示す赤いラインは、ネットワークがブレークするポイントを示します。このポイントは、ハードウェアおよびソフトウェアの動作方法に関する情報に基づいて決まります。 ネットワーク負荷を意味する緑のラインは、新しいアプリケーションの追加やその他の要因により、ネットワークの負荷が自然に増えていく状態を示します。

ベースラインの目標は、次の事項を確認することです。

- ネットワークが緑のラインのどの位置にいるのか。
- ネットワーク負荷の増加の速度。
- 可能であればこの 2 つの項目が交差する時点を予測する。

ベースラインを定期的に実行することで、現在の状態を確認し、さらに障害発生時点を推定し、そのような障害に対して事前に対策をとることができます。 さらに、ネットワークのアップグレードについて、予算額をいつ、どこで、どう使用するかを、情報に基づいて決定できます。

なぜベースラインなのか

ベースライン プロセスにより、ネットワーク上での重大なリソース制限の問題を特定し、このような問題を適切に考慮できます。 これらの問題は、コントロール プレーン リソースまたはデータ プレーン リソースとして説明できます。 コントロール プレーン リソースは、デバイス内の特定のプラットフォームとモジュールに固有であり、次のようなさまざまな問題の影響を受ける可能性があります。

- データの使用状況
- 有効な機能
- ネットワーク設計

次のパラメータをはじめとするコントロール プレーン リソース：

- CPU 使用率
- メモリ使用率
- バッファ使用率

リンク使用率やバックプレーン使用率などのデータ プレーン リソースは、トラフィックのタイプ

と量の影響だけを受けます。 重要な領域のリソース使用率のベースラインを実行することで、パフォーマンスの重大な問題やネットワーク メルトダウンを回避できます。

音声やビデオなどの遅延の影響を受けやすいアプリケーションの導入により、ベースラインはより重要になっています。 一般的な Transmission Control Protocol/Internet Protocol（TCP/IP）アプリケーションは、一定の遅延を許容できます。 一方、音声とビデオのアプリケーションの場合は、User Datagram Protocol（UDP）に基づいており、再転送やネットワークの輻輳を許容しません。

新たなアプリケーションの組み合わせに伴い、ベースラインにより、コントロール プレーンとデータ プレーンの両方のリソース使用率に関する問題を理解し、継続的な正常運用のための変更とアップグレードをプロアクティブに計画することができます。

データ ネットワークは長年にわたり使用されてきました。 最近まで、ネットワークを稼動することは、多少の エラーも比較的許容されていました。 Voice over IP（VoIP）などの遅延の影響を受けやすいアプリケーションが急速に受け入れられるにつれて、ネットワークの稼動は、より困難になるとともに高い精度が要求されています。 精度を高め、ネットワーク管理者にネットワーク管理のための強固な基盤を提供するには、ネットワークがどのように運用されているかを理解しておくことが重要です。 そのためには、ベースラインと呼ばれるプロセスを導入する必要があります。

ベースライン目標

ベースラインの目標は、次の事項を実行することです。

1. ネットワーク デバイスの現在の状態を判別する。
2. その状態を標準パフォーマンス ガイドラインと比較する。
3. ネットワーク装置の状況がガイドラインを超えたら、警告するようにしきい値を設定する。

データ分析に必要なデータの量と時間から、プロセスについて容易に理解できるようにするため、最初にベースライン範囲を制限する必要があります。 ネットワークのコア部分から開始するのが、最も合理的で、通常、最も有益です。 ネットワークのこの部分は、通常、最も小さく、しかも最高の安定性が求められるためです。

このドキュメントでは説明を簡潔にするため、1 つの非常に重要な Simple Network Management Protocol 管理情報ベース（SNMP MIB）である cpmCPUTotal5min のベースライン実行方法を説明します。 cpmCPUTotal5min は、Cisco ルータの中央処理装置（CPU）の 5 分間の減衰平均であり、コントロール プレーンのパフォーマンス インジケータです。 このベースラインは、Cisco 7000 シリーズのルータで実行します。

プロセスを学習し終えたら、大規模な SNMP データベース内の任意のデータに適用できます。次のような SNMP データベースは、ほとんどの Cisco 装置に用意されています。

- 統合サービス デジタル網（ISDN）の使用状況
- 非同期転送モード（ATM）セル損失
- システム メモリの空き容量

コア ベースライン フローチャート

次のフローチャートでは、コア ベースライン プロセスの基本的なステップを示しています。 これらのステップを実行するときに使える製品やツールも用意されていますが、柔軟性や使いやすさの点で差があります。 ネットワーク管理システム（NMS）ツールを使用してベースラインを実

行することを予定している場合でも、プロセスについて習得し、ネットワークが実際にどのように動作しているのかを理解しておくことが推奨されます。 ほとんどのツールは本質的には同じことをするため、このプロセスにより、いくつかの NMS ツールの動作方法に関する疑問が解決する場合があります。

ベースライン手順

ステップ 1： ハードウェア、ソフトウェアおよび設定インベントリのコンパイル

ハードウェア、ソフトウェア、および設定のインベントリを作成しておくことは、さまざまな理由から非常に重要です。 まず、場合によっては Cisco SNMP MIB は実行中の Cisco IOS リリースに固有です。 MIB オブジェクトによっては、新しいものと交換されたり、また完全に破棄されます。 データ収集後には、ハードウェアのインベントリが最も重要です。これは、最初のベースライン実行後に、Cisco 装置上の CPU のタイプ、メモリ容量などに基づいてしきい値を設定する場合が多いためです。 構成インベントリも、現在の構成を確認する意味で重要です。 ベースラインを実行してバッファの調整などを行ってから、装置の構成を変更する場合もあります。

Cisco ネットワークの場合、ベースラインのステップ 1 の部分を最も効率的に実行するには、CiscoWorks2000 Resource Manager Essentials（Essentials）を使用します。 このソフトウェアがネットワークに正しくインストールされている場合は、Essentials のデータベースにすべてのデバイスの最新インベントリが保存されています。 問題の有無を確認するには、そのインベントリを参照するだけで済みます。

次の表は、Essentials からエクスポートされた Cisco Router Class ソフトウェアのインベントリレポートを Microsoft Excel で編集した例です。 このインベントリから、Cisco IOS リリース 12.0x および 12.1x で検出された SNMP MIB データとオブジェクト ID（OID）を使用する必要があることがわかります。

Device Name	ルータのタイプ	バージョン	[Software Version]
field-2500a.embu-mlab.cisco.com	Cisco 2511	M	12.1(1)
qdm-7200.embu-mlab.cisco.com	Cisco 7204	B	12.1(1)E
voip-3640.embu-mlab.cisco.com	Cisco 3640	0x00	12.0(3c)
wan-1700a.embu-mlab.cisco.com	Cisco 1720	0x101	12.1(4)
wan-2500a.embu-mlab.cisco.com	Cisco 2514	L	12.0(1)
wan-3600a.embu-mlab.cisco.com	Cisco 3640	0x00	12.1(3)
wan-7200a.embu-mlab.cisco.com	Cisco 7204	B	12.1(1)E
172.16.71.80	Cisco 7204	B	12.0(5T)

Essentials がネットワークにインストールされていない場合は、UNIX ワークステーションから UNIX コマンド ライン ツール **snmpwalk** を使用して IOS バージョンを検索できます。 これを次の例で示します。 このコマンドがどのように動作するかわからない場合は、UNIX プロンプトで **man snmpwalk** と入力し、詳細情報を確認します。 どの MIB OID をベースラインにするかを選択する場合、IOS バージョンが重要になります。これは MIB オブジェクトが IOS に依存しているためです。 また、ルータ タイプを把握することで、CPU、バッファなどについて使用するしきい値を決定できることにも注意してください。

```
nsahpov6% snmpwalk -v1 -c private 172.16.71.80 system
system.sysDescr.0 : DISPLAY STRING- (ascii): Cisco Internetwork Operating System Software
IOS (tm) 7200 Software (C7200-JS-M), Version 12.0(5)T, RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2001 by cisco Systems, Inc.
Compiled Fri 23-Jul-2001 23:02 by kpma
system.sysObjectID.0 : OBJECT IDENTIFIER:
.iso.org.dod.internet.private.enterprises.cisco.ciscoProducts.cisco7204
```

ステップ 2： SNMP MIB がルータでサポートされていることを確認する

ベースラインのためにポーリングするデバイスのインベントリが存在するため、ポーリングする特定の OID を選択できます。 これにより、必要なデータが実際に存在するかどうかを事前に検証する場合のフラストレーションが大幅に解消されます。 cpmCPUTotal5min MIB オブジェクトは、CISCO-PROCESS-MIB にあります。

ポーリングする OID を探すには、Cisco の CCO ウェブ サイトにある変換テーブルが必要です。 Web ブラウザからこの Web サイトにアクセスするには、Cisco MIB ページにアクセスして、OID のリンクをクリックします。

FTP サーバからこの Web サイトにアクセスするには、**ftp://ftp.cisco.com/pub/mibs/oid/** と入力します。 このサイトから、デコードされ OID 番号によりソートされた特定の MIB をダウンロードできます。

次の例に、CISCO-PROCESS-MIB.oid テーブル一部を示します。 次の例では、cpmCPUTotal5minMIB の OID が .1.3.6.1.4.1.9.9.109.1.1.1.1.5 であることがわかります。

注: OID の先頭に「.」を 必ず追加してください。このようにしないと、そのポーリング試行時にエラーが発生します。 また、OID をインスタンスにするには、OID の最後に「.1」を追加する必要があります。 これによって、探している OID のインスタンスが装置に伝えられます。 ルータに複数の CPU が搭載されている場合などに、OID には特定タイプのデータのインスタンスが複数含まれていることがあります。

```
ftp://ftp.cisco.com/pub/mibs/oid/CISCO-PROCESS-MIB.oid
### THIS FILE WAS GENERATED BY MIB2SCHEMA
"org" "1.3"
"dod" "1.3.6"
"internet" "1.3.6.1"
"directory" "1.3.6.1.1"
"mgmt" "1.3.6.1.2"
"experimental" "1.3.6.1.3"
"private" "1.3.6.1.4"
"enterprises" "1.3.6.1.4.1"
"cisco" "1.3.6.1.4.1.9"
"ciscoMgmt" "1.3.6.1.4.1.9.9"
"ciscoProcessMIB" "1.3.6.1.4.1.9.9.109"
"ciscoProcessMIBObjects" "1.3.6.1.4.1.9.9.109.1"
"ciscoProcessMIBNotifications" "1.3.6.1.4.1.9.9.109.2"
"ciscoProcessMIBConformance" "1.3.6.1.4.1.9.9.109.3"
"cpmCPU" "1.3.6.1.4.1.9.9.109.1.1"
```

```
"cpmProcess" "1.3.6.1.4.1.9.9.109.1.2"
"cpmCPUTotalTable" "1.3.6.1.4.1.9.9.109.1.1.1"
"cpmCPUTotalEntry" "1.3.6.1.4.1.9.9.109.1.1.1.1"
"cpmCPUTotalIndex" "1.3.6.1.4.1.9.9.109.1.1.1.1.1"
"cpmCPUTotalPhysicalIndex" "1.3.6.1.4.1.9.9.109.1.1.1.1.2"
"cpmCPUTotal5sec" "1.3.6.1.4.1.9.9.109.1.1.1.1.3"
"cpmCPUTotal1min" "1.3.6.1.4.1.9.9.109.1.1.1.1.4"
"cpmCPUTotal5min" "1.3.6.1.4.1.9.9.109.1.1.1.1.5"
```

MIB OID が使用可能であり、動作していることを確認するために、MIB OID をポーリングする 場合は 2 つの一般的な方法があります。 一括データ収集の開始前にポーリングすることを推奨します。これにより、存在しないデータのポーリングに時間を費やし、空のデータベースが作成されることがなくなります。 MIB OID をポーリングする 1 つの方法に、HP OpenView Network Node Manager (NNM) 、または CiscoWorks Windows などの NMS プラットフォームから MIB ウォーカを使用し、チェックする OID を入力する方法があります。

次に、HP OpenView SNMP MIB ウォーカの例を示します。

MIB OID を容易にポーリングする別の方法として、次の例に示すように、UNIX コマンド **snmpwalk** を使用する方法もあります。

```
nsahpov6% cd /opt/OV/bin
nsahpov6% snmpwalk -v1 -c private 172.16.71.80 .1.3.6.1.4.1.9.9.109.1.1.1.1.5.1

cisco.ciscoMgmt.ciscoProcessMIB.ciscoProcessMIBObjects.cpmCPU.cpmCPUTotalTable.cpmCPUTotalEntry.
cpmCPUTotal5min.1 : Gauge32: 0
```

両方の例で MIB から値 0 が戻されていますが、これは、そのポーリング サイクルでは CPU の平均使用率が 0 %であることを示します。 デバイスが正しいデータで応答するようにすることが難しい場合は、そのデバイスに対して ping を実行し、Telnet でデバイスにアクセスしてください。それでもまだ問題がある場合には、SNMP 設定と SNMP コミュニティ文字列を調べてください。 問題を解決するには、別の MIB または IOS の他のバージョンを探す必要がある場合があります。

ステップ 3： ルータから特定の SNMP MIB オブジェクトへポーリングと記録を行う

MIB オブジェクトをポーリングして、その出力結果を記録する方法がいくつかあります。 市販製品、シェアウェア製品、スクリプト、およびベンダー ツールを使用できます。 すべてのフロントエンド ツールは SNMP **get** プロセスを使用して情報を取得します。 主に、構成の柔軟性とデータをデータベースに記録する方法が異なります。 プロセッサ MIB を調べ、これらのさまざまな方法がどのように機能するかを確認します。

OID がルータでサポートされていることが判明したので、次にポーリングの頻度と記録方法を決定する必要があります。 CPU MIB を 5 分間隔でポーリングすることを推奨します。 この間隔が短いと、ネットワークまたはデバイスに対する負荷が増大します。MIB 値は 5 分間平均であるため、平均値よりも頻繁にポーリングする操作は有用ではありません。 ベースライン ポーリングでは少なくとも 2 週間の期間を設定することが一般に推奨されます。これにより、ネットワークでの週単位のビジネス サイクルを少なくとも 2 つ分析できます。

次の画面では、HP OpenView Network Node Manager version 6.1 によって MIB オブジェクトを追加する方法を示しています。 メイン画面から [Options] > [Data Collection & Thresholds] を選択します。

次に [Edit] > [Add] > [MIB Objects] を選択します。

メニューから OID ストリングを追加して [Apply] をクリックします。 これで、HP OpenView プラットフォームに MIB オブジェクトが入力されたので、ポーリングすることができます。

次に、この OID について、ポーリングするルータを HP OpenView に指示する必要があります。

[Data Collection] メニューから [Edit] > [Add] > [MIB Collections] を選択します。

[Source] フィールドに、ポーリング対象ルータのドメイン ネーム システム（DNS）名または IP アドレスを入力します。

Set Collection Mode リストから Store, No Thresholds を選択します。

[Polling Interval] を [5m]（5 分間隔）に設定します。

[Apply] をクリックします。

変更を有効にするため、[File] > [Save] を選択する必要があります。

収集が正しく設定されていることを検証するには、ルータの収集サマリー行を強調表示し、[Actions] > [Test SNMP] を選択します。 この結果、コミュニティ ストリングが正確であるかどうか、および OID の全インスタンスがポーリング対象となっているかどうかをチェックできます。

[Close] をクリックして、1 週間にわたり収集を実行します。 週期間の終了時に、分析データを取得します。

データを分析しやすくするには、ASCII ファイルにデータをダンプして、Microsoft Excel などの表計算ツールにインポートします。 HP OpenView NNM を使用して同じことを実行する場合、コマンドライン ツール **snmpColDump** を使用できます。 設定された各収集は、/var/opt/OV/share/databases/snmpCollect/ ディレクトリ内のファイルに書き込まれます。

次のコマンドを使用して ASCII ファイル **testfile** にデータを抽出します。

```
snmpColDump /var/opt/OV/share/databases/snmpCollect/cpmCPUTotal5min.1 > testfile
```

注: cpmCPUTotal5min.1 は、OID のポーリング開始時に HP OpenView NNM が作成したデータベース ファイルです。

作成されたテスト ファイルは、次の例のように表示されます。

```
03/01/2001 14:09:10 nsa-gw.cisco.com 1
03/01/2001 14:14:10 nsa-gw.cisco.com 1
03/01/2001 14:19:10 nsa-gw.cisco.com 1
03/01/2001 14:24:10 nsa-gw.cisco.com 1
03/01/2001 14:29:10 nsa-gw.cisco.com 1
03/01/2001 14:34:10 nsa-gw.cisco.com 1
03/01/2001 14:39:10 nsa-gw.cisco.com 1
03/01/2001 14:44:10 nsa-gw.cisco.com 1
03/01/2001 14:49:10 nsa-gw.cisco.com 1
03/01/2001 14:54:10 nsa-gw.cisco.com 1
03/01/2001 14:59:10 nsa-gw.cisco.com 1
03/……
```

テスト ファイルの出力結果が UNIX 端末に表示されたら、File Transfer Protocol（FTP）を使用して各自の PC に転送できます。

自分のスクリプトを使用してもデータを収集できます。 そのためには、5 分ごとに CPU OID に対して **snmpget** コマンドを実行して、結果を .csv ファイルにダンプします。

ステップ 4： しきい値を判別するためにデータを分析する

データが収集されたため、データの分析を開始されます。 この段階のベースラインでは、しきい値を設定します。しきい値を設定することで、パフォーマンスや障害を正確に測定でき、しかもしきい値のモニタリングをオンにしておくと、アラームが発生し過ぎません。 最も簡単な設定方法の 1 つは、データを Microsoft Excel などの表計算にインポートしてから、散布図にプロットする方法です。 この方法を使用すると、特定のしきい値についてある装置を監視している場合に、その装置が例外アラートを生成する回数を簡単に確認できます。 ベースラインを実行せずにしきい値をオンにする操作は推奨されません。このように操作すると、選択したしきい値を超えたデバイスからアラート ストームが発生する可能性があるためです。

テスト ファイルを Excel スプレッドシートにインポートするには、Excel を開き、[File] > [Open] を選択し、データ ファイルを選択します。

次に、ファイルのインポート中に、Excel アプリケーションから確認を求められます。

完了すると、インポートされたファイルは次の画面のように表示されます。

散布図では、さまざまなしきい値設定がネットワークでどのように機能しているかをより簡単に視覚化できます。

散布図を作成するには、インポートしたファイルの列 C を強調表示させ、[Chart Wizard] アイコンをクリックします。 次に、Chart Wizard のステップに従って、散布図を作成します。

次の図に示す [Chart Wizard] のステップ 1 で、[Standard Types] タブを選択し、[XY (Scatter)] グラフ タイプを選択します。 次に [Next] をクリックします。

次の図に示す [Chart Wizard] のステップ 2 で、[Data Range] タブを選択し、データ範囲と [Columns] オプションを選択します。 [Next] をクリックします。

次の図に示す [Chart Wizard] のステップ 3 で、グラフのタイトルと、X 軸および Y 軸の値を入力し、[Next] をクリックします。

[Chart Wizard] のステップ 4 で、散布図を新しいページに作成するか、または既存のページにオブジェクトとして作成するかを選択します。

[Finish] をクリックし、希望する位置にグラフを配置します。

"「What if」 分析

これで、散布図を使用して分析できます。 ただし続行する前に、次の事項を確認しておく必要があります。

- ベンダーが推奨する MIB 変数のしきい値は何か（この例では、シスコがベンダー）。一般に、コア ルータが平均 CPU 使用率の 60 %を超えないようにすることを推奨します。 シスコが 60 % を選択したのは、ルータにトラブルが発生したり、ネットワークに障害が起こった場合に、ルータに多少のオーバーヘッドが必要だからです。 シスコでは、ルーティング プロトコルを再計算または再設定する必要が生じる場合に備えてコア ルータに必要な CPU オーバーヘッドは、約 40 %であると推定しています。 このパーセント値は、使用しているプロトコル、およびネットワークのトポロジや安定性によっても変わってきます。
- しきい値の設定に 60 % を使用したらどうなるか。散布図で 60 で横に線をひくと、60 % の CPU 使用率を超えるデータ ポイントがないことがわかります。 そのため、NMS 端末でしき

い値を 60 に設定すると、ポーリングの間、しきい値のアラートが発生しません。 このルータでは 60 % は許容可能です。 ただし散布図では一部のデータ ポイントが 60 に近いことがわかります。 ルータがしきい値である 60 % に近づく時点を把握できると便利です。これにより、CPU が 60 % に近づいていることを事前に把握し、60 % に達した時点での処理を計画することができます。
- しきい値を 50 % に設定したらどうなるか。このルータがこのポーリング サイクルで 50 % の使用率に 4 回到達し、到達するたびにしきい値アラームが生成されると推定されます。 異なるしきい値を設定したらどうなるかを確認するために*ルータ グループ*を表示させる場合、このプロセスの重要度が増します。 たとえば、「しきい値をコア ネットワーク全体の 50 % に設定したらどうなるか。」などです。 しきい値を 1 つだけ選択することが難しいことがわかります。

CPU しきい値の「What If」分析

しきい値を容易に設定する方法として、レディ、セット、ゴーのしきい値方法があります。 この方法論では、連続して 3 つのしきい値を使用します。

- レディ（Ready）：将来どのデバイスが注意を必要とする可能性が高いかを予測するために設定したしきい値
- セット（Set）：初期インジケータとして使用するしきい値。このしきい値によって、修理、再設定、またはアップグレードの計画を開始するように警告されます。
- ゴー（Go）：ユーザまたはベンダーが障害状況であると考えるしきい値。修理するには何らかのアクションが必要です。 この例では 60 % です。

次の表では、レディ、セット、ゴーの戦略について示しています。

しきい値（Threshold）	Action	結果
45 %	さらに調査する	アクション プランのオプションのリスト
50 %	アクションプランの策定	アクション プランのステップのリスト
60 %	アクション プランを実装する	ルータではしきい値の超過がなくなった。 レディ モードに戻る

レディ、セット、ゴー方法論により、前述したオリジナルのベースライン チャートが変わります。 次の図は、変更後のベースライン チャートを示しています。 グラフで他の交点を特定できる場合は、これまでよりも多くの時間を計画と対応に費やすことができます。

このプロセスでは、ネットワークでの例外に重点が置かれ、その他のデバイスについては注意が払われていない点に注意してください。 デバイスは、しきい値を下回っている限り正常であると想定されます。

これらの手順を最初から実行している場合は、ネットワークの健全性を適切に維持できます。 このタイプの計画を実行することは、予算を計画する上でも有効です。 上位 5 件の**ゴー（go）ルータ**、中間の**セット（set）ルータ**、下位の**レディ（ready）ルータ**が判明している場合は、ルータの種類と、アクション プラン オプションの内容に基づいて、アップグレードに必要な予算を容易に計画できます。 同じ戦略を Wide-Area Network（WAN）やその他の MIB OID にも使用できま

す。

ステップ 5： 修正によって認識される差し迫った問題

ステップ 5 は、ベースライン プロセスの中でも簡単な部分の 1 つです。 どのデバイスがゴーしきい値を超えたかを特定できれば、そのデバイスをしきい値以下に戻すアクション プランを作成します。

利用できるオプションについて、Cisco's Technical Assistance Center（TAC）までご相談いただくか、システム エンジニアにお問い合せください。 しきい値を下げることに費用がかかるとは考えないでください。 CPU の問題によっては、設定を変更してすべてのプロセスを効率的な方法で確実に実行すれば、解決できます。 たとえば一部のアクセス コントロール リスト（ACL）により、パケットがルータを通過するパスが原因でルータの CPU の使用率が非常に高くなることがあります。 場合によっては、パケット スイッチング パスを変更し、CPU に対する ACL の影響を軽減するために NetFlow スイッチングを実装できます。 問題の内容を問わず、このステップでは全ルータのしきい値を設定値以下に戻すことが必要です。これにより、しきい値アラームが発生し過ぎて NMS 端末がフラッディングするリスクを負わずに、後でしきい値を設定できます。

ステップ 6： しきい値モニタリングをテストする

このステップでは、プロダクション ネットワークで使用するツールにより、ラボでしきい値をテストします。 しきい値をモニタリングするには、一般的に、2 つの方法があります。 自分のネットワークに最適な方法を選択する必要があります。

- SNMP プラットフォームまたはその他の SNMP モニタリング ツールによって、ポーリングと比較を行う方法この方法では、トラフィックをポーリングするのにネットワークの帯域幅をより多く使用し、SNMP プラットフォームでのポーリング サイクルにも時間がかかります。
- ルータの Remote Monitoring（RMON; リモート モニタリング）アラームおよびイベントの設定によって、しきい値が超過した場合だけアラートが送られる方法この方法を使用すると、ネットワーク帯域幅の使用状況は減りますが、ルータでのメモリと CPU の利用率が増えます。

SNMP を使用したしきい値の実装

HP OpenView NNM を使用した SNMP 方式を設定するには、初期ポーリングのセットアップの場合と同様に [Options] > [Data Collection & Thresholds] を選択します。 ただし今回は、収集メニューで Store, No Thresholds を選択する代わりに、Store, Check Thresholds を選択します。 しきい値の設定後に、ルータに複数の ping または複数の SNMP ウォークを送信すると、ルータの CPU 使用率を上昇させることができます。 意図的に CPU の利用率がしきい値を超えるようにできない場合は、しきい値を下げる必要がある場合もあります。 いずれの場合でも、しきい値メカニズムが機能していることを確認する必要があります。

この方法の使用に関する制限の 1 つは、複数のしきい値を同時に設定できないことです。 3 つの異なるしきい値を同時に設定するには、3 つの SNMP プラットフォームが必要になります。 Concord Network Health や T rinagy TREND などのツールでは、同一 OID インスタンスで複数のしきい値を設定できます。

システムで一度に処理できるしきい値が 1 つだけの場合は、レディ、セット、ゴー方式を順次使

用することを検討できます。 つまり、継続的に**レディ**しきい値に達する場合、調査を開始して、そのデバイスのセット レベルまでしきい値を上げるということです。 継続的に**セット レベル**に達する場合は、アクション プランを策定し、そのデバイス**ゴー** レベルにまでしきい値を上げます。 さらに、ゴーのしきい値に頻繁に達しているときは、アクション プランを実行します。 この方法は、同時に 3 つのしきい値を設定する方法と同じように動作しますが、 SNMP プラットフォームしきい値設定の変更にはこれよりも多少時間がかかります。

RMON アラームおよびイベントを使用したしきい値の実装

RMON アラームとイベントの設定を使用すると、ルータ自体が複数のしきい値を監視できます。ルータはしきい値が超過している状態を検知すると、SNMP プラットフォームに SNMP トラップを送信します。 トラップを転送するためには、各自のルータ設定に SNMP トラップ受信装置を設定する必要があります。 アラームとイベントとの間には相互関係があります。 アラームは、特定のしきい値について OID をチェックします。 しきい値に達すると、アラーム プロセスによりイベント プロセスが起動し、イベント プロセスでは SNMP トラップ メッセージを送信するか、RMON ログ エントリを作成するか、またはこの両方を実行できます。 このコマンドの詳細については、「RMON アラームおよびイベント設定コマンド」を参照してください。

次のルータ設定コマンドを実行すると、ルータは 300 秒ごとに cpmCPUTotal5min を監視します。 CPU が 60 % を超えるとイベント 1 が発生し、CPU が 40 % に戻るとイベント 2 が発生します。 いずれの場合も、SNMP トラップ メッセージがコミュニティ プライベート ストリングと共に NMS ステーションに送信されます。

レディ、セット、ゴーの方法を使用するには、次の設定文をすべて使用します。

```
rmon event 1 trap private description "cpu hit60%" owner jharp
rmon event 2 trap private description "cpu recovered" owner jharp
rmon alarm 10 cpmCPUTotalTable.1.5.1 300 absolute rising 60 1 falling 40 2 owner jharp


rmon event 3 trap private description "cpu hit50%" owner jharp
rmon event 4 trap private description "cpu recovered" owner jharp
rmon alarm 20 cpmCPUTotalTable.1.5.1 300 absolute rising 50 3 falling 40 4 owner jharp

rmon event 5 trap private description "cpu hit 45%" owner jharp
rmon event 6 trap private description "cpu recovered" owner jharp
rmon alarm 30 cpmCPUTotalTable.1.5.1 300 absolute rising 45 5 falling 40 6 owner jharp
```

次の例では、上記ステートメントによって設定された **show rmon alarm コマンド**の出力を示します。

```
zack#sh rmon alarm Alarm 10 is active, owned by jharp Monitors cpmCPUTotalTable.1.5.1 every 300
second(s) Taking absolute samples, last value was 0 Rising threshold is 60, assigned to event 1
Falling threshold is 40, assigned to event 2 On startup enable rising or falling alarm Alarm 20
is active, owned by jharp Monitors cpmCPUTotalTable.1.5.1 every 300 second(s) Taking absolute
samples, last value was 0 Rising threshold is 50, assigned to event 3 Falling threshold is 40,
assigned to event 4 On startup enable rising or falling alarm Alarm 30 is active, owned by jharp
Monitors cpmCPUTotalTable.1.5.1 every 300 second(s) Taking absolute samples, last value was 0
Rising threshold is 45, assigned to event 5 Falling threshold is 40, assigned to event 6 On
startup enable rising or falling alarm
```

次の例では、**show rmon event コマンド**の出力を示します。

```
zack#sh rmon event Event 1 is active, owned by jharp Description is cpu hit60% Event firing
causes trap to community private, last fired 00:00:00 Event 2 is active, owned by jharp
Description is cpu recovered Event firing causes trap to community private, last fired 02:40:29
Event 3 is active, owned by jharp Description is cpu hit50% Event firing causes trap to
community private, last fired 00:00:00 Event 4 is active, owned by jharp Description is cpu
recovered Event firing causes trap to community private, last fired 00:00:00 Event 5 is active,
```

```
owned by jharp Description is cpu hit 45% Event firing causes trap to community private, last
fired 00:00:00 Event 6 is active, owned by jharp Description is cpu recovered Event firing
causes trap to community private, last fired 02:45:47
```

この両方の方法を試して、どちらの方法が自分の環境に最適であるかを確認します。 両方の方法を組み合わせて順調に動作する場合もあります。 いずれの場合も、すべてが正しく動作していることを確認するため、ラボ環境でテストを実施する必要があります。 ラボでのテスト完了後に、少数のルータでの限られた導入環境で、オペレーション センターへのアラートの送信プロセスをテストできます。

この場合、プロセスをテストするためにしきい値を低くする必要があります。 実働しているルータ上で人為的に CPU 稼働率を上げることは推奨しません。 また、アラートがオペレーション センターの NMS 端末に着信すると、装置がしきい値を超過したときに確実に通知される報告ポリシーが存在することを確認する必要があります。 この設定は、Cisco IOS バージョン 12.1(7)を使用して、ラボで試験を行いました。 問題が発生する場合は、シスコのエンジニアリングまたはシステム エンジニアに連絡し、ご使用の IOS バージョンで不具合があるかどうかを確認します。

ステップ 7： SNMP または RMON を使用したしきい値モニタリングの実施

ラボ、または限定された実装の中で、しきい値モニタリングのテストが完全に終わったら、コアネットワーク全体にしきい値を設定する準備ができています。 これで、バッファ、空いているメモリ、Cyclic Redundancy Check (CRC; サイクリック冗長性検査) エラー、ATM のセル損失などをはじめとする、ネットワーク内にある他の重要な MIB 変数について、このベースライン プロセスを体系的に実行できます。

RMON アラームとイベントの設定を使用している場合、NMS ステーションからのポーリングを停止できます。 この結果、NMS サーバ上の負荷が減り、ネットワーク上のポーリング データ量も低減します。 重要なネットワーク正常性インジケータを確認するためこのプロセスを体系的に実行することで、ネットワーク機器が RMON アラームとイベントを使用して機器自体をモニタする状況を容易に実現できます。

追加 MIB

このプロセスについて習得したら、ベースラインを実行しモニタする他の MIB を調査することができます。 次のサブセクションで、いくつかの役に立つ OID の簡単なリストと説明を示します。

ルータの MIB

メモリの特性は、ルータの正常性を確認する上で非常に役立ちます。 正常なルータではほぼ常に、処理に使用可能なバッファ領域があります。 ルータのバッファ領域が不足し始めると、新しいバッファを作成し、着信パケットと発信パケットのためのバッファを検出するために、CPU の使用率が上昇します。 バッファに関する詳細な解説は、このドキュメントの対象外です。 ただし、一般原則として、正常なルータではバッファ ミスはほとんど発生せず、バッファ障害や、空きメモリがゼロになる状況が発生してはなりません。

オブジェクト	説明	OID
ciscoMemoryPoolFree	管理対象装置で現在、未使用のメモリ プールのバイト数	1.3.6.1.4.1.9.9.48.1.1.1.6
ciscoMemoryPoolLargestFree	現在、未使用のメモリ プールの隣接最大バイト	1.3.6.1.4.1.9.9.48.1.1.1.7

	数	
bufferElMiss	バッファ要素のミス数	1.3.6.1.4.1.9.2.1.12
bufferFail	バッファ割り当て失敗数	1.3.6.1.4.1.9.2.1.46
bufferNoMem	空きメモリ不足によるバッファの作成障害数	1.3.6.1.4.1.9.2.1.47

Catalyst スイッチ MIB

オブジェクト	説明	OID
cpmCPUTotal5min	最後の 5 分間に CPU がビジー状態であった合計パーセント。 このオブジェクトは、OLD-CISCO-SYSTEM-MIB の avgBusy5 オブジェクトを推奨しない。	1.3.6.1.4.1.9.9.109.1.1.1.5
cpmCPUTotal5sec	最後の 5 秒間に CPU がビジー状態であった合計パーセント。 このオブジェクトは、OLD-CISCO-SYSTEM-MIB の busyPer オブジェクトを廃棄する。	1.3.6.1.4.1.9.9.109.1.1.1.3
sysTraffic	前回のポーリング期間の帯域利用率のパーセント	1.3.6.1.4.1.9.5.1.1.8
sysTrafficPeak	最後にポート カウンタがクリアーされてから、またはシステムが起動してからのトラフィック メータの最高値	1.3.6.1.4.1.9.5.1.1.19
sysTrafficPeaktime	メータ最高値の発生後の時間（100 分の 1 秒単位）	1.3.6.1.4.1.9.5.1.1.20
portTopNUtilization	システムのポート利用率	1.3.6.1.4.1.9.5.1.20.2.1.4
portTopNBufferOverFlow	システムのポートのバッファ オーバーフロー数	1.3.6.1.4.1.9.5.1.20.2.1.10

シリアル リンク MIB

オブジェクト	説明	OID
locIfInputQueueDrops	入力キューがいっぱいのためドロップされたパケット数	1.3.6.1.4.1.9.2.2.1.1.26
locIfOutputQueueDrops	出力キューがいっぱいのためドロップされたパケット数	1.3.6.1.4.1.9.2.2.1.1.27

locIfInCRC	冗長性チェックサム エラーが発生した入力パケット数	1.3.6.1.4.1.9.2.2.1.1.12

RMON アラームおよびイベント設定コマンド

アラーム

RMON アラームは、次の構文によって設定できます。

rmon alarm *number variable interval* **{delta | absolute} rising-threshold** *value* [*event-number*] **falling-threshold** *value* [*event-number*] [**owner** *string*]

要素	説明
番号を入力します	アラーム番号。RMON MIB のアラーム テーブル内にある alarmIndex と同一である。
variable	監視対象の MIB オブジェクト。RMON MIB のアラーム テーブルで使用される alarmVariable に変換する。
間隔	アラームが MIB 変数を監視する時間（秒単位）。RMON MIB のアラーム テーブルで使用される alarmInterval と同一である。
delta	MIB 変数間の変更をテストする。RMON MIB のアラーム テーブルの alarmSampleType に影響を与える。
absolute	各 MIB 変数を直接、テストする。RMON MIB のアラーム テーブルの alarmSampleType に影響を与える。
rising-threshold value	アラームが起動する値
event-number	（オプション）上昇しきい値または下降しきい値が、制限値を超えたときに起動するイベント番号。 この値は、RMON MIB のアラーム テーブルの alarmRisingEventIndex または the alarmFallingEventIndex と同一である。
falling-threshold value	アラームがリセットされる値
owner string	（オプション）アラームの所有者を示す。RMON MIB のアラーム テーブルの alarmOwner と同一である。

イベント

RMON イベントは、次の構文によって設定できます。

rmon event *number* [**log**] [**trap** *community*] [**description** *string*] [**owner** *string*]

要素	説明
番号を入力します	割り当てられているイベント番号。これは、RMON MIB の eventTable 内にある eventIndex と同一である。
log	（オプション）イベントが起動し、RMON MIB の eventType を log または log-and-trap に設定するときに、RMON ログ エントリを作成する。
trap community	（オプション）このトラップに使用される SNMP コミュニティ ストリング。 この行の RMON MIB の eventType の設定を snmp-trap または log-and-trap に設定します。 この値は、RMON MIB の eventTable 内の eventCommunityValue と同一である。
description string	（オプション）イベントの説明を示す。RMON MIB のアラーム テーブルのイベントの説明と同一である。
owner string	（オプション）このイベントの所有者を示す。RMON MIB のアラーム テーブルの eventOwner と同一である。

RMON アラームとイベントの実装

RMON アラームとイベントの実装の詳細については、『*ネットワーク管理システムのベスト プラクティス*』ホワイト ペーパーの「RMON アラームとイベントの実装」の項を参照してください。

関連情報

- テクニカルサポートとドキュメント - シスコ