

Baseline Process Best Practice 백서

목차

[소개](#)

[초기 계획](#)

[기준이란?](#)

[베이스라인이 필요한 이유](#)

[초기 계획 목표](#)

[핵심 기준 순서도](#)

[초기 계획 절차](#)

[1단계: 하드웨어, 소프트웨어 및 구성 인벤토리 컴파일](#)

[2단계: SNMP MIB가 라우터에서 지원되는지 확인합니다.](#)

[3단계: 라우터에서 특정 SNMP MIB 개체 폴링 및 기록](#)

[4단계: 데이터 분석을 통해 임계값 결정](#)

[5단계: 확인된 즉각적인 문제 해결](#)

[6단계: 테스트 임계값 모니터링](#)

[7단계: SNMP 또는 RMON을 사용하여 임계값 모니터링 구현](#)

[추가 MIB](#)

[라우터 MIB](#)

[Catalyst 스위치 MIB](#)

[직렬 링크 MIB](#)

[RMON Alarm 및 Event Configuration 명령](#)

[경보](#)

[이벤트](#)

[RMON 경보 및 이벤트 구현](#)

[관련 정보](#)

소개

이 문서에서는 고가용성 네트워크에 대한 베이스라인 개념 및 절차에 대해 설명합니다. 여기에는 성공을 평가하는 데 도움이 되는 네트워크 베이스라인 설정 및 임계값에 대한 중요한 성공 요인이 포함되어 있습니다. 또한 Cisco HAS(High Availability Services) 팀이 파악한 모범 사례 지침을 따르는 기본 및 임계값 프로세스와 구현에 대한 중요한 세부 정보도 제공합니다.

이 문서에서는 베이스라인 적용 프로세스를 단계별로 안내합니다. 일부 현재 NMS(Network Management System) 제품은 이 프로세스를 자동화하는 데 도움이 될 수 있지만, 자동화 툴을 사용한 수동 툴을 사용한 베이스라인 프로세스는 동일합니다. 이러한 NMS 제품을 사용하는 경우 고유한 네트워크 환경에 대한 기본 임계값 설정을 조정해야 합니다. 이러한 임계값이 의미 있고 정확하도록 지능적으로 선택하는 프로세스가 중요합니다.

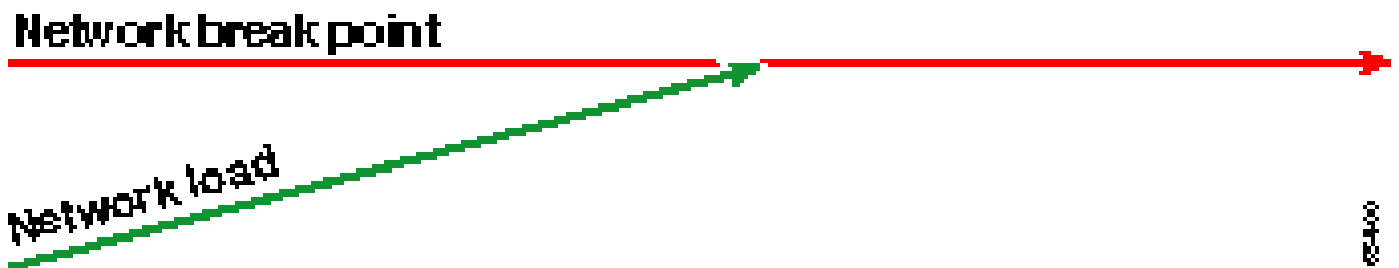
초기 계획

기준이란?

기준선은 네트워크가 설계대로 작동하는지 확인하기 위해 정기적으로 네트워크를 조사하는 프로세스입니다. 특정 시점의 네트워크 상태를 자세히 설명하는 단일 보고서가 아닙니다. 베이스라인 프로세스에 따라 다음 정보를 얻을 수 있습니다.

- 하드웨어 및 소프트웨어의 상태에 대한 유용한 정보 확보
- 네트워크 리소스의 현재 사용률 확인
- 네트워크 경보 임계값에 대한 정확한 결정
- 현재 네트워크 문제 파악
- 향후 문제 예측

기준선을 보는 또 다른 방법은 다음 다이어그램에 나타나 있다.



빨간색 선, 즉 네트워크 중단점은 네트워크가 중단되는 지점이며, 이는 하드웨어와 소프트웨어의 성능을 알고 있기 때문에 결정됩니다. 녹색 선, 즉 네트워크 로드는 새로운 애플리케이션이 추가됨에 따라 네트워크에서 발생하는 부하의 자연스러운 진행 및 기타 요인입니다.

기준선의 목적은 다음을 결정하는 것입니다.

- 네트워크의 현재 위치
- 네트워크 부하 증가 속도
- 바라건대 두 사람이 어느 시점에서 교차하게 될지 예상해 보길 바란다

기준선을 정기적으로 수행해 실패가 발생할 현재 상태를 파악하고 미리 대비할 수 있다. 또한 네트워크 업그레이드에 예산을 사용할 시기, 장소 및 방법에 대한 정보에 입각한 결정을 내리는 데 도움이 됩니다.

베이스라인이 필요한 이유

기본 프로세스는 네트워크에서 중요한 리소스 제한 문제를 식별하고 적절히 계획하는 데 도움이 됩니다. 이러한 문제는 컨트롤 플레인 리소스 또는 데이터 플레인 리소스로 설명할 수 있습니다. 컨트롤 플레인 리소스는 디바이스 내의 특정 플랫폼 및 모듈에 고유하며 다음과 같은 여러 문제의 영향을 받을 수 있습니다.

- 데이터 사용률

- 활성화된 기능
- 네트워크 설계

컨트롤 플레인 리소스에는 다음과 같은 매개변수가 포함됩니다.

- CPU 사용률
- 메모리 사용률
- 버퍼 사용률

데이터 플레인 리소스는 트래픽의 유형 및 양에 의해서만 영향을 받으며 링크 사용률 및 백플레인 사용률을 포함합니다. 중요 영역에 대한 리소스 사용률을 베이스라인으로 지정하여 심각한 성능 문제 또는 더 심각한 네트워크 중단을 방지할 수 있습니다.

음성 및 비디오와 같이 레이턴시에 민감한 애플리케이션이 도입됨에 따라 이제는 베이스라인 설정이 그 어느 때보다 중요해졌습니다. 기존의 TCP/IP(Transmission Control Protocol/Internet Protocol) 애플리케이션은 이를 용납하고 일정 수준의 지연을 허용합니다. 음성 및 비디오는 UDP(User Datagram Protocol)를 기반으로 하며 재전송이나 네트워크 정체를 허용하지 않습니다.

새로운 애플리케이션 혼합으로 인해 베이스라인을 구성하면 컨트롤 플레인 및 데이터 플레인 리소스 활용 문제를 모두 파악하고 변경 및 업그레이드를 사전에 계획하여 지속적인 성공을 보장할 수 있습니다.

데이터 네트워크는 여러 해 동안 존재해 왔습니다. 최근까지도 네트워크를 계속 운영하는 것은 꽤 용납할 만한 프로세스였으며 약간의 오류 여지도 있었습니다. VoIP(Voice over IP)와 같이 레이턴시에 민감한 애플리케이션의 수용이 증가함에 따라 네트워크 실행 작업이 점점 더 어려워지고 있으며 더 정밀한 작업이 요구되고 있습니다. 네트워크 관리자가 네트워크를 관리할 수 있는 기반을 좀 더 정확하게 파악하려면 네트워크가 어떻게 실행되고 있는지 파악하는 것이 중요합니다. 이를 위해서는 기준선이라는 과정을 거쳐야 한다.

초기 계획 목표

기준 요소 목표는 다음과 같습니다.

1. 네트워크 디바이스의 현재 상태 확인
2. 해당 상태를 표준 성과 지침과 비교
3. 상태가 해당 지침을 초과할 경우 알림을 보내도록 임계값을 설정합니다

많은 양의 데이터와 데이터를 분석하는 데 걸리는 시간 때문에 프로세스를 더 쉽게 배울 수 있도록 먼저 기준 요소의 범위를 제한해야 합니다. 가장 논리적이고 때로는 가장 유익한 출발점이 네트워크의 핵심입니다. 네트워크의 이 부분은 일반적으로 가장 작고 가장 안정적입니다.

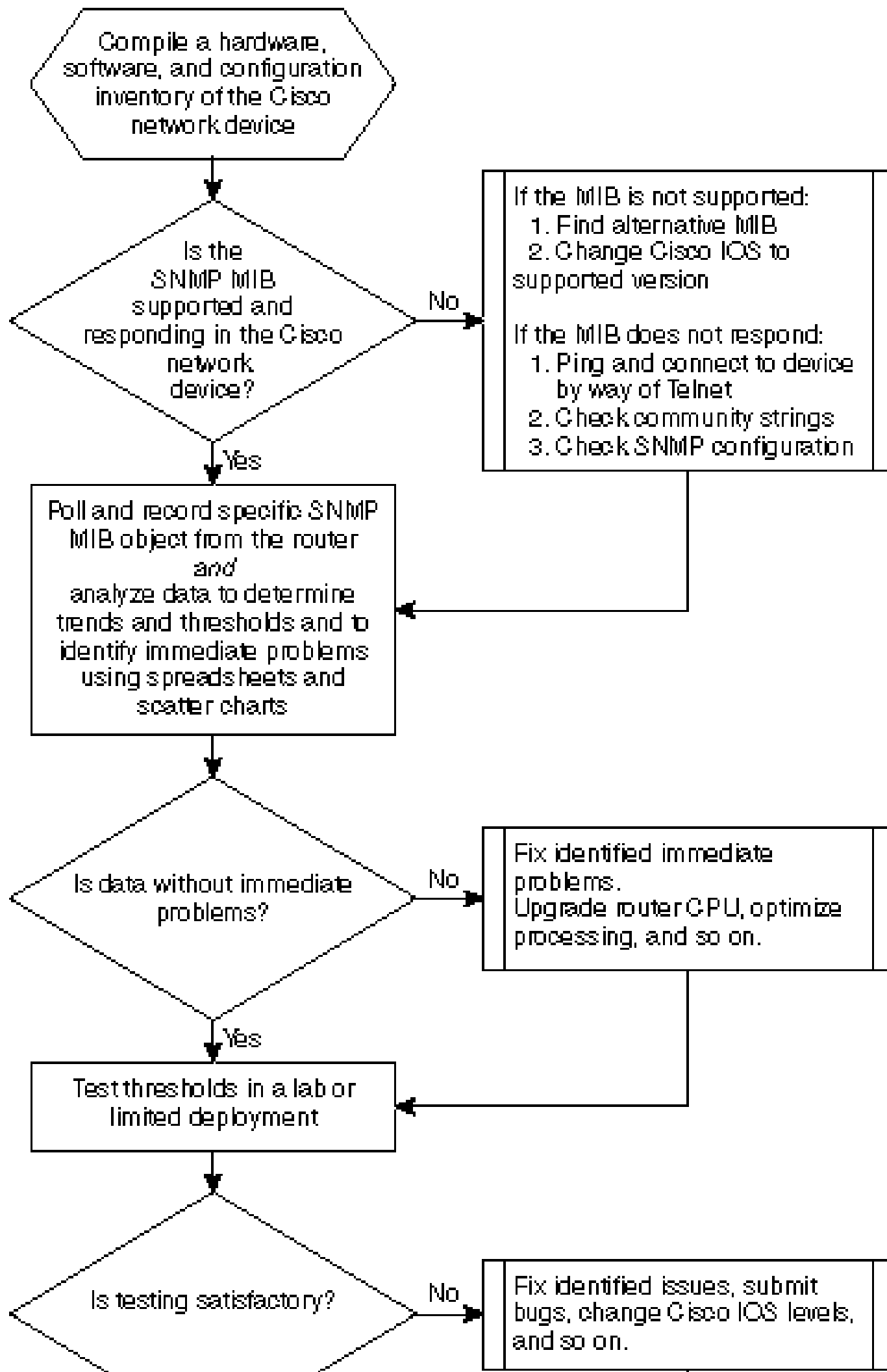
이 문서에서는 간소화를 위해 매우 중요한 SNMP MIB(Simple Network Management Protocol Management Information Base)의 기준을 설정하는 방법을 설명합니다. cpmCPUTotal5min. cpmCPUTotal5min은 Cisco 라우터의 CPU(Central Processing Unit)의 5분 감소 평균이며, 컨트롤 플레인 성능 지표입니다. 기준선은 Cisco 7000 Series 라우터에서 수행됩니다.

프로세스를 학습한 후에는 다음과 같이 대부분의 Cisco 디바이스에서 사용할 수 있는 방대한 SNMP 데이터베이스에서 사용 가능한 모든 데이터에 적용할 수 있습니다.

- ISDN(Integrated Services Digital Network) 사용
- ATM(Asynchronous Transfer Mode) 셀 손실
- 사용 가능한 시스템 메모리

핵심 기준 순서도

다음 순서도는 핵심 기준 프로세스의 기본 단계를 보여줍니다. 이러한 단계를 수행할 수 있는 제품 및 툴이 있지만 유연성이나 사용 편의성에 문제가 있는 경우가 많습니다. NMS(Network Management System) 툴을 사용하여 베이스라인 작업을 수행하려는 경우에도 이 방법을 사용하면 프로세스를 공부하고 네트워크가 실제로 작동하는 방식을 이해하는 데 도움이 됩니다. 대부분의 툴이 기본적으로 동일한 작업을 수행하므로 이 프로세스는 또한 일부 NMS 툴의 작동 방식에 대한 수 수깨끼를 일부 제거할 수 있습니다.



초기 계획 절차

1단계: 하드웨어, 소프트웨어 및 구성 인벤토리 컴파일

여러 가지 이유로 하드웨어, 소프트웨어 및 컨피그레이션의 인벤토리를 컴파일하는 것이 매우 중요합니다. 먼저, Cisco SNMP MIB는 경우에 따라 실행 중인 Cisco IOS 릴리스에만 적용됩니다. 일부 MIB 객체는 새 객체로 교체되거나 완전히 제거됩니다. 초기 베이스라인 이후에 설정해야 하는 임계값은 Cisco 디바이스의 CPU 유형, 메모리 양 등을 기반으로 하는 경우가 많기 때문에 데이터를 수집한 후에는 하드웨어 인벤토리가 가장 중요합니다. 컨피그레이션 인벤토리는 현재 컨피그레이션을 파악하는 데에도 중요합니다. 베이스라인 이후 버퍼를 조정하기 위해 디바이스 컨피그레이션을 변경하는 등의 작업을 수행할 수 있습니다.

Cisco 네트워크의 기준에서 이 부분을 수행하는 가장 효율적인 방법은 CiscoWorks2000 Resource Manager Essentials(Essentials)를 사용하는 것입니다. 이 소프트웨어가 네트워크에 올바르게 설치된 경우 Essentials에는 데이터베이스에 있는 모든 디바이스의 현재 인벤토리가 있어야 합니다. 재고를 살펴봐야 문제가 있는지 알 수 있습니다.

다음 표는 Essentials에서 내보낸 다음 Microsoft Excel에서 편집한 Cisco Router Class 소프트웨어 인벤토리 보고서의 예입니다. 이 인벤토리에서는 12.0x 및 12.1x Cisco IOS 릴리스에 있는 SNMP MIB 데이터 및 OID(Object Identifier)를 사용해야 합니다.

| 디바이스 이름 | 라우터 유형 | 버전 | 소프트웨어 버전 |
|---------------------------------|------------|-------|-----------|
| 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 | 12.0조(5조) |

Essentials가 네트워크에 설치되어 있지 않은 경우 UNIX 워크스테이션에서 UNIX 명령줄 도구 snmpwalk를 사용하여 IOS 버전을 찾을 수 있습니다. 다음 예에 나와 있습니다. 이 명령의 작동 방

식을 모를 경우 UNIX 프롬프트에 man snmpwalk를 입력하여 자세한 내용을 확인하십시오. IOS 버전은 MIB 객체가 IOS에 종속되므로 베이스라인에 사용할 MIB OID를 선택할 때 중요합니다. 또한 라우터 유형을 파악하면 나중에 CPU, 버퍼 등에 대해 어떤 임계값이 필요한지 결정할 수 있습니다.

```
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
```

2단계: SNMP MIB가 라우터에서 지원되는지 확인합니다.

베이스라인에 대해 폴링할 디바이스의 인벤토리가 있으므로 폴링할 특정 OID를 선택할 수 있습니다. 사전에 원하는 데이터가 실제로 있는지 확인한다면 많은 좌절감을 줄일 수 있습니다. cpmCPUTotal5min MIB 객체는 CISCO-PROCESS-MIB에 있습니다.

폴링할 OID를 찾으려면 Cisco CCO 웹 사이트에서 사용할 수 있는 변환 테이블이 필요합니다. 웹 브라우저에서 이 웹 사이트에 액세스하려면 [Cisco MIBs 페이지](#)로 이동하여 OID 링크를 클릭합니다.

FTP 서버에서 이 웹 사이트에 액세스하려면 ftp://ftp.cisco.com/pub/mibs/oid/을 입력합니다. 이 사이트에서 OID 번호별로 디코딩 및 정렬된 특정 MIB를 다운로드할 수 있습니다.

다음 예는 CISCO-PROCESS-MIB.oid 테이블에서 추출됩니다. 이 예에서는 cpmCPUTotal5min MIB의 OID가 .1.3.6.1.4.1.9.9.109.1.1.1.5임을 보여 줍니다.

참고: OID의 시작 부분에 "."를 추가하는 것을 잊지 마십시오. 그렇지 않으면 폴링하려고 할 때 오류가 발생합니다. OID의 끝에 ".1"을 추가하여 인스턴스화해야 합니다. 이는 찾고 있는 OID의 인스턴스를 디바이스에 알려줍니다. 라우터에 여러 CPU가 있는 경우와 같이 OID에는 특정 데이터 유형의 인스턴스가 두 개 이상 있는 경우도 있습니다.

<#root>

```
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"
```

```

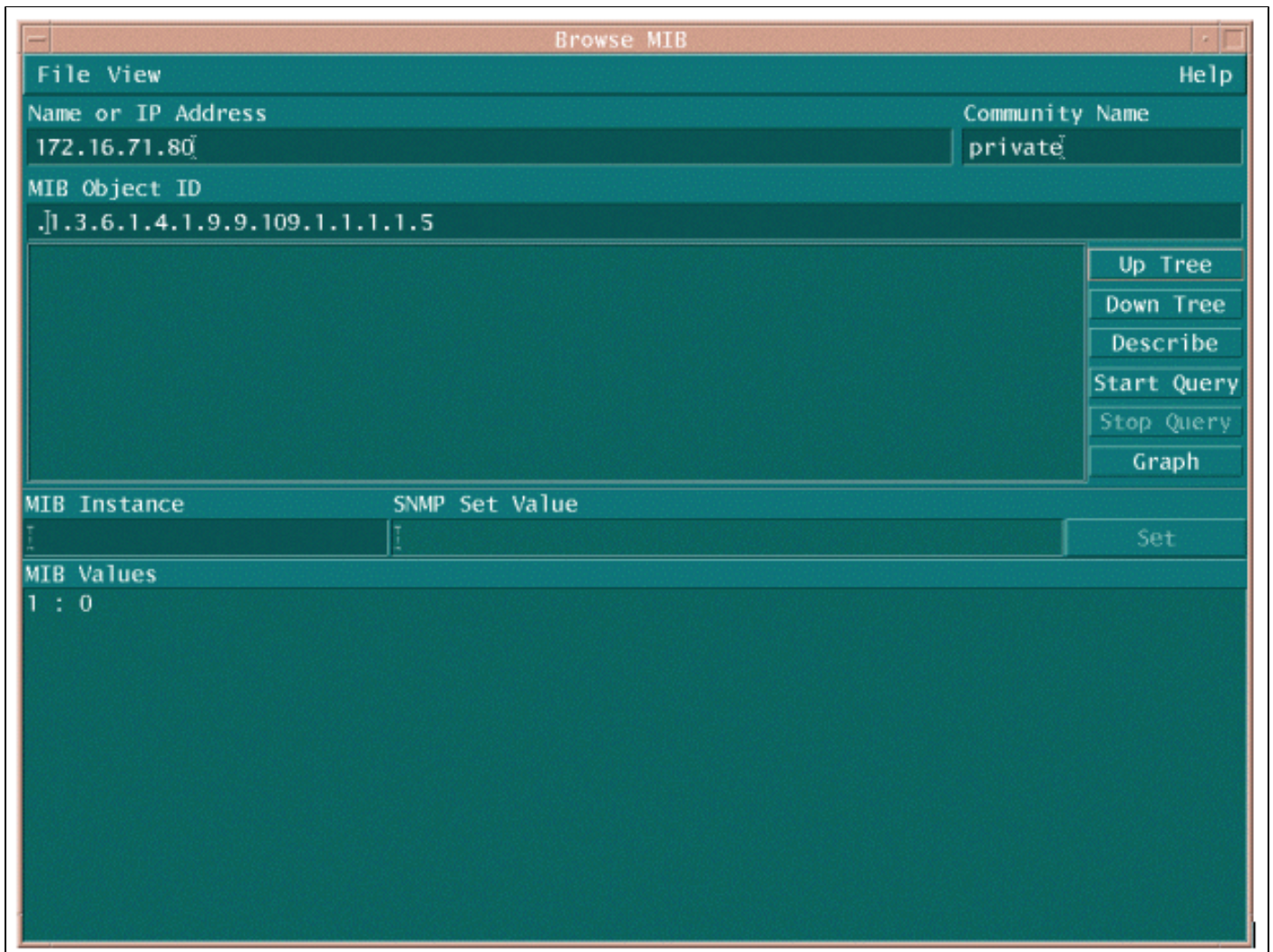
"cisكوProcessMIBConformance" "1.3.6.1.4.1.9.9.109.3"
"cpmCPU" "1.3.6.1.4.1.9.9.109.1.1"
"cpmProcess" "1.3.6.1.4.1.9.9.109.1.2"
"cpmCPUTotalTable" "1.3.6.1.4.1.9.9.109.1.1.1"
"cpmCPUTotalEntry" "1.3.6.1.4.1.9.9.109.1.1.1.1"
"cpmCPUTotalIndex" "1.3.6.1.4.1.9.9.109.1.1.1.1.1"
"cpmCPUTotalPhysicalIndex" "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 OID가 사용 가능하고 작동하는지 확인하기 위해 MIB OID를 폴링하는 두 가지 일반적인 방법이 있습니다. 대량 데이터 수집을 시작하기 전에 이 작업을 수행하는 것이 좋습니다. 그렇게 하면 없는 항목을 폴링하여 빈 데이터베이스로 끝내는 데 시간을 낭비하지 않습니다. 이렇게 하는 한 가지 방법은 HP OpenView NNM(Network Node Manager) 또는 CiscoWorks Windows와 같은 NMS 플랫폼의 MIB 워커를 사용하여 확인할 OID를 입력하는 것입니다.

다음은 HP OpenView SNMP MIB walker의 예입니다.



MIB OID를 폴링하는 또 다른 쉬운 방법은 다음 예에 표시된 것처럼 UNIX 명령 snmpwalk를 사용하는 것입니다.


```
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.cpmCPU
```

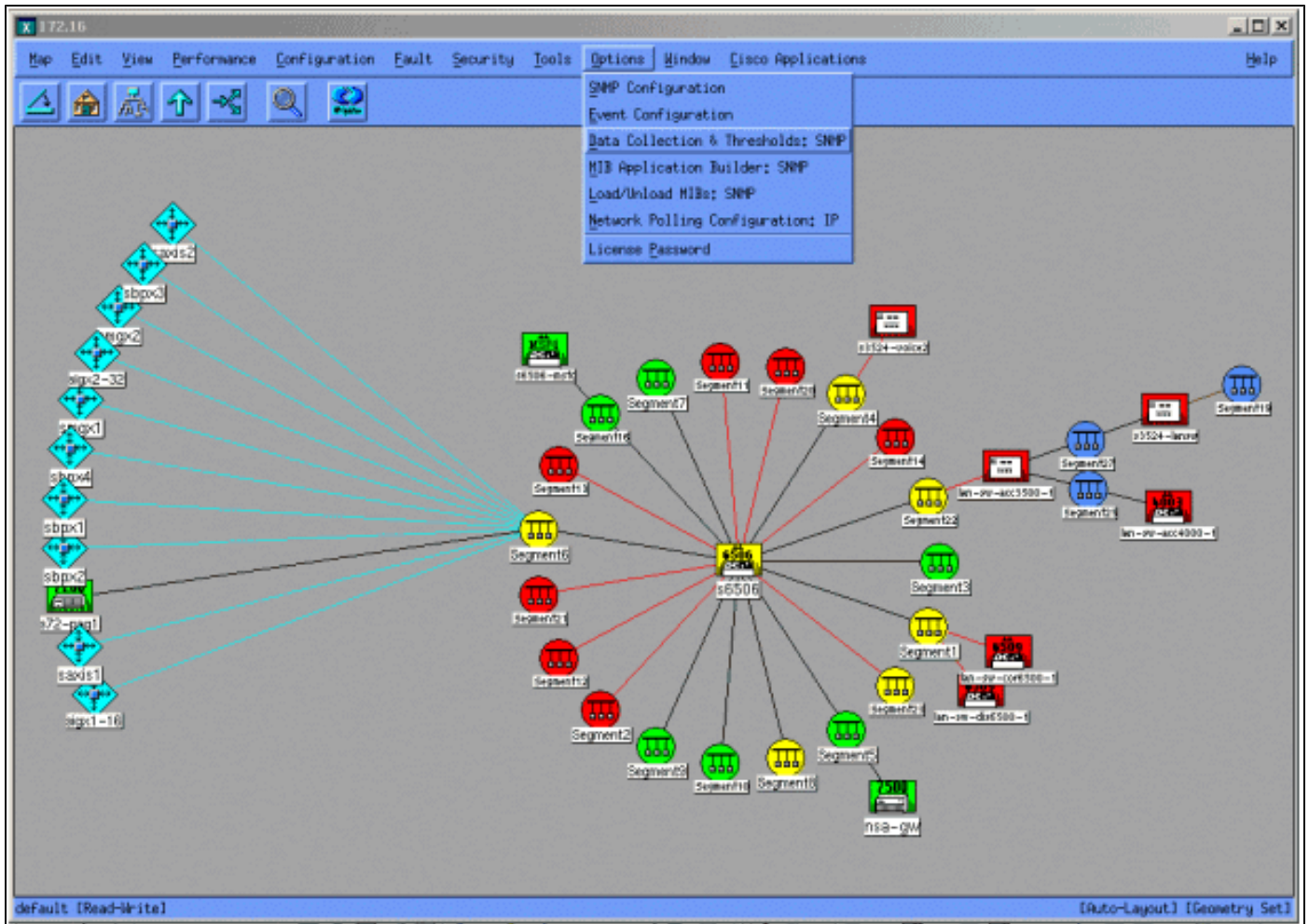
두 예에서 MIB는 0의 값을 반환했으며, 이는 해당 폴링 주기의 CPU 사용률이 평균적으로 0%임을 의미합니다. 장치가 올바른 데이터로 응답하도록 하는 데 문제가 있는 경우 장치에 ping을 시도하고 텔넷을 통해 장치에 액세스하십시오. 여전히 문제가 있는 경우 SNMP 컨피그레이션 및 SNMP 커뮤니티 문자열을 확인합니다. 이 작업을 수행하려면 대체 MIB 또는 다른 버전의 IOS를 찾아야 할 수도 있습니다.

3단계: 라우터에서 특정 SNMP MIB 객체 폴링 및 기록

MIB 객체를 폴링하고 출력을 기록하는 방법에는 여러 가지가 있습니다. 기성 제품, 쉘어웨어 제품, 스크립트 및 벤더 툴을 사용할 수 있습니다. 모든 프론트엔드 툴은 SNMP 가져오기 프로세스를 사용하여 정보를 얻습니다. 주요 차이점은 구성의 융통성 및 데이터가 데이터베이스에 기록되는 방법에 있습니다. 다시, 프로세서 MIB를 참조하여 이러한 다양한 방법이 어떻게 작동하는지 확인하십시오.

이제 라우터에서 OID가 지원된다는 것을 알았으므로 폴링할 빈도와 기록 방법을 결정해야 합니다. CPU MIB는 5분 간격으로 폴링하는 것이 좋습니다. 간격이 낮으면 네트워크 또는 디바이스의 로드가 증가하며, MIB 값이 아무리 낮더라도 5분 평균이므로 평균된 값보다 자주 폴링하는 것은 유용하지 않습니다. 또한 일반적으로 기본 폴링에는 최소 2주의 기간이 있으므로 네트워크에서 매주 최소 2회의 비즈니스 주기를 분석할 수 있습니다.

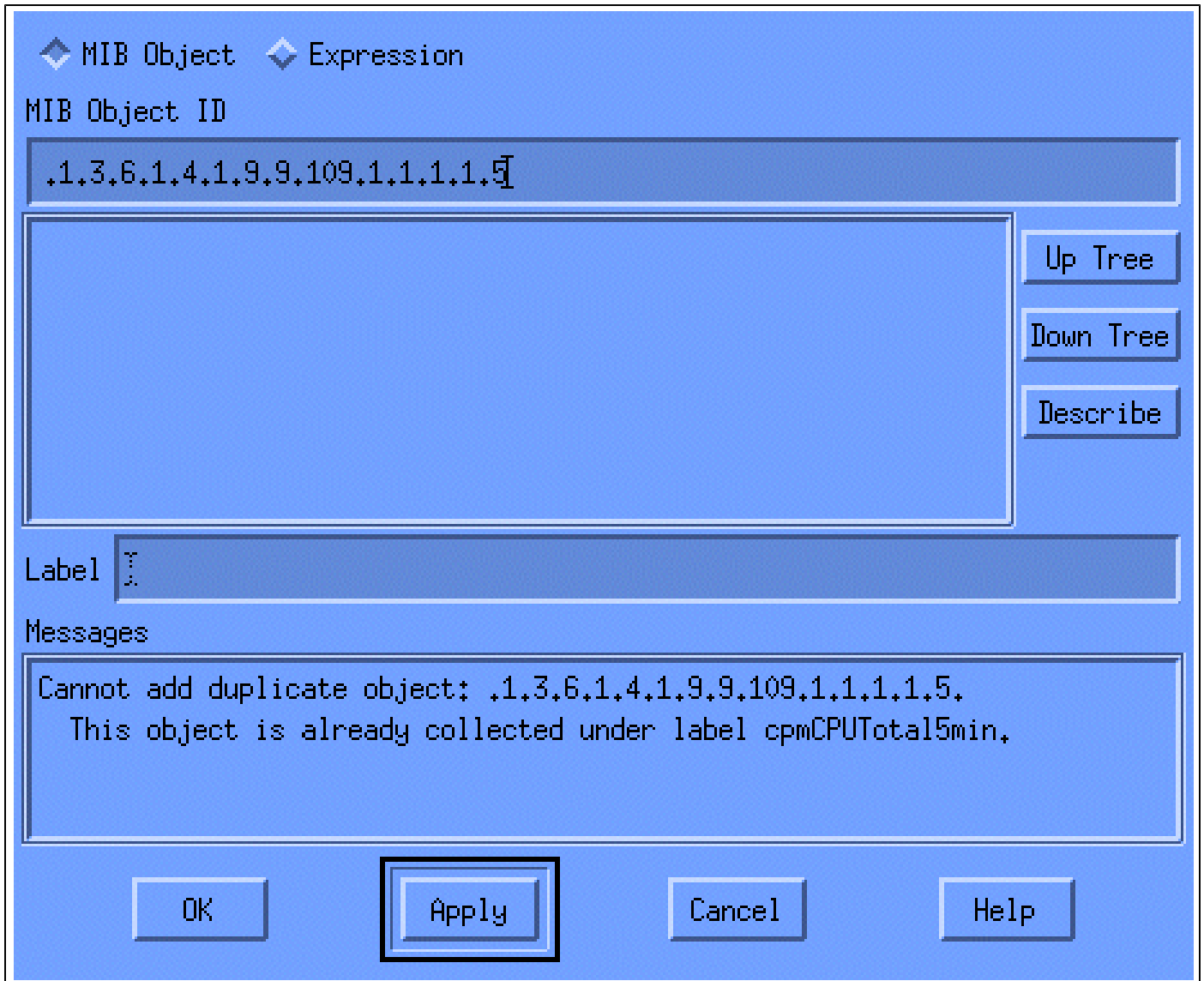
다음 화면에서는 HP OpenView Network Node Manager 버전 6.1에서 MIB 객체를 추가하는 방법을 보여줍니다. 메인 화면에서 옵션 > 데이터 수집 및 임계값을 선택합니다.



그런 다음 Edit(편집) > Add(추가) > MIB Objects(MIB 개체)를 선택합니다.

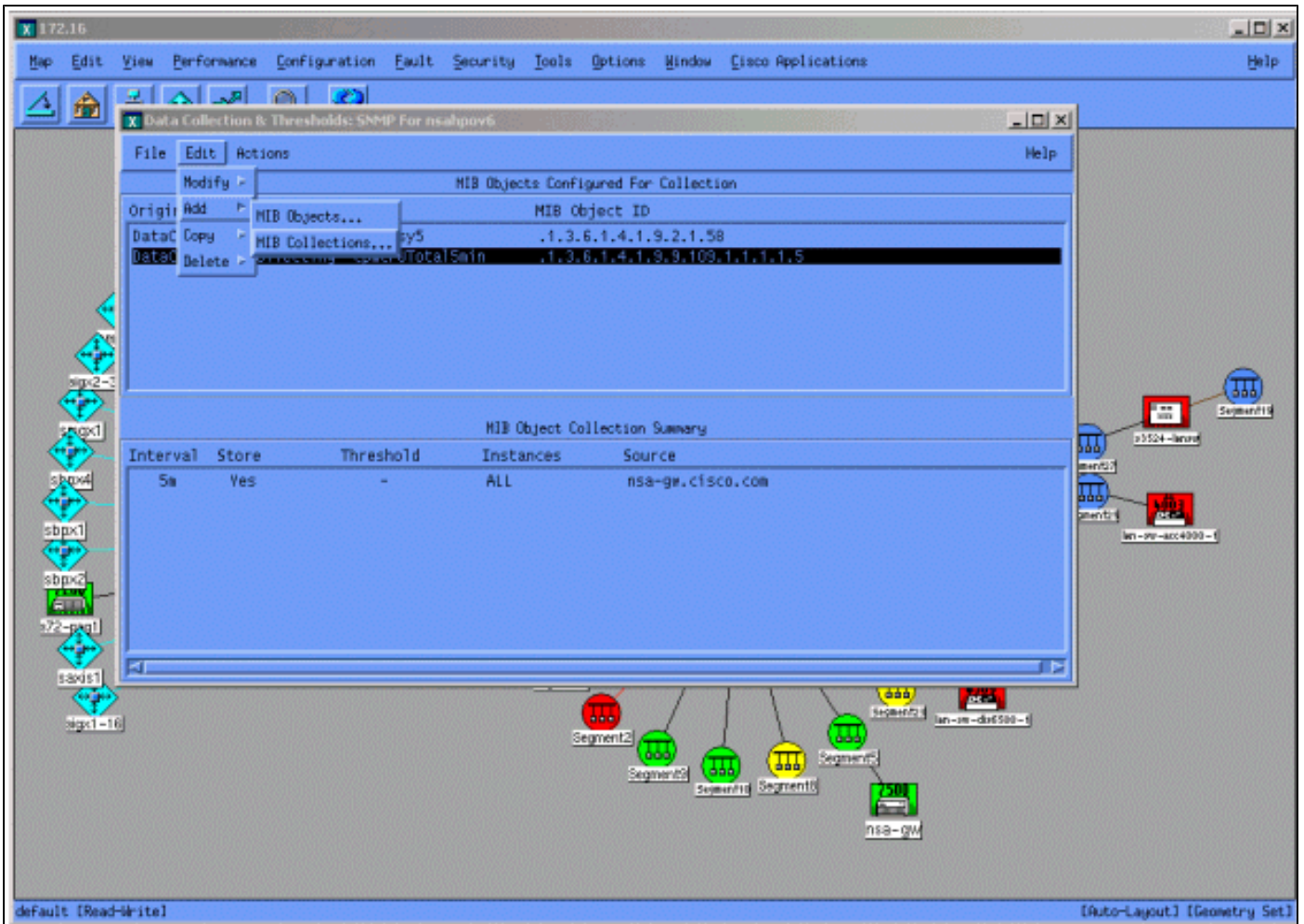


메뉴에서 OID 문자열을 추가하고 Apply를 클릭합니다. 이제 MIB 개체를 폴링할 수 있도록 HP OpenView 플랫폼에 MIB 개체를 입력했습니다.



그런 다음 HP OpenView에 이 OID에 대해 폴링할 라우터를 알려줘야 합니다.

Data Collection(데이터 수집) 메뉴에서 Edit(편집) > Add(추가) > MIB Collections(MIB 수집)를 선택합니다.

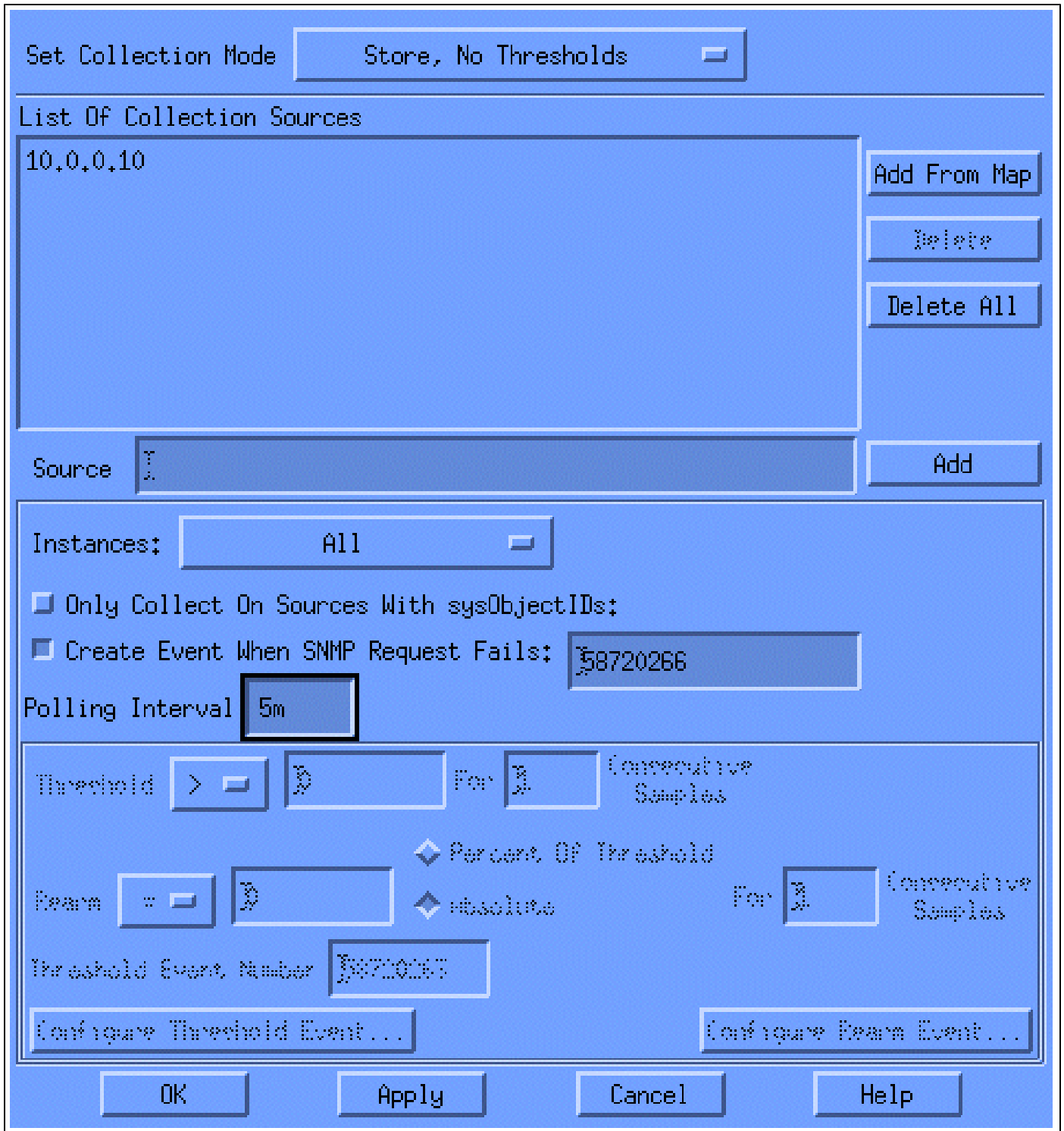


Source 필드에 폴링할 라우터의 DNS(Domain Naming System) 이름 또는 IP 주소를 입력합니다.

Set Collection Mode 목록에서 Store, No Thresholds를 선택합니다.

Polling Interval(폴링 간격)을 5m(5분 간격)으로 설정합니다.

적용을 클릭합니다.



변경 사항을 적용하려면 파일 > 저장 을 선택해야 합니다.

컬렉션이 제대로 설정되었는지 확인하려면 라우터에 대한 컬렉션 요약 행을 강조 표시하고 Actions(작업) > Test SNMP(SNMP 테스트)를 선택합니다. 커뮤니티 문자열이 올바른지 확인하고 OID의 모든 인스턴스를 폴링합니다.

```
Starting SNMP test for all instances on nsa-gw.cisco.com.  
Checking MIB .1.3.6.1.4.1.9.9.109.1.1.1.1.5:
```

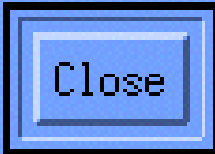
```
.1.3.6.1.4.1.9.9.109.1.1.1.1.5 (instance 1): 0  
.1.3.6.1.4.1.9.9.109.1.1.1.1.5 (instance 2): 1  
.1.3.6.1.4.1.9.9.109.1.1.1.1.5 (instance 3): 1
```

```
Tested all instances.
```

```
Instances which will be collected:
```

```
1 2 3
```

```
All instances will be collected.
```

A rectangular button with a double border and the word "Close" centered inside.

닫기를 클릭하고 일주일 동안 컬렉션을 실행합니다. 주별 기간이 끝나면 분석용 데이터를 추출합니다.

데이터를 ASCII 파일로 덤프하고 Microsoft Excel과 같은 스프레드시트 도구로 가져오면 더 쉽게 분석할 수 있습니다. HP OpenView NNM에서 이를 수행하려면 명령줄 도구인 snmpColDump를 사용할 수 있습니다. 구성된 각 컬렉션은 /var/opt/OV/share/databases/snmpCollect/ 디렉토리의 파일에 씁니다.

다음 명령을 사용하여 testfile이라는 ASCII 파일로 데이터를 추출합니다.

```
<#root>
```

```
snmpColDump /var/opt/OV/share/databases/snmpCollect/cpmCPUtotal5min.1 >
```

```
testfile
```

참고: cpmCPUtotal5min.1은 OID 폴링이 시작될 때 HP OpenView NNM에서 생성한 데이터베이스 파일입니다.

생성된 테스트 파일은 다음 예와 유사하게 표시됩니다.

```
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/.....
```

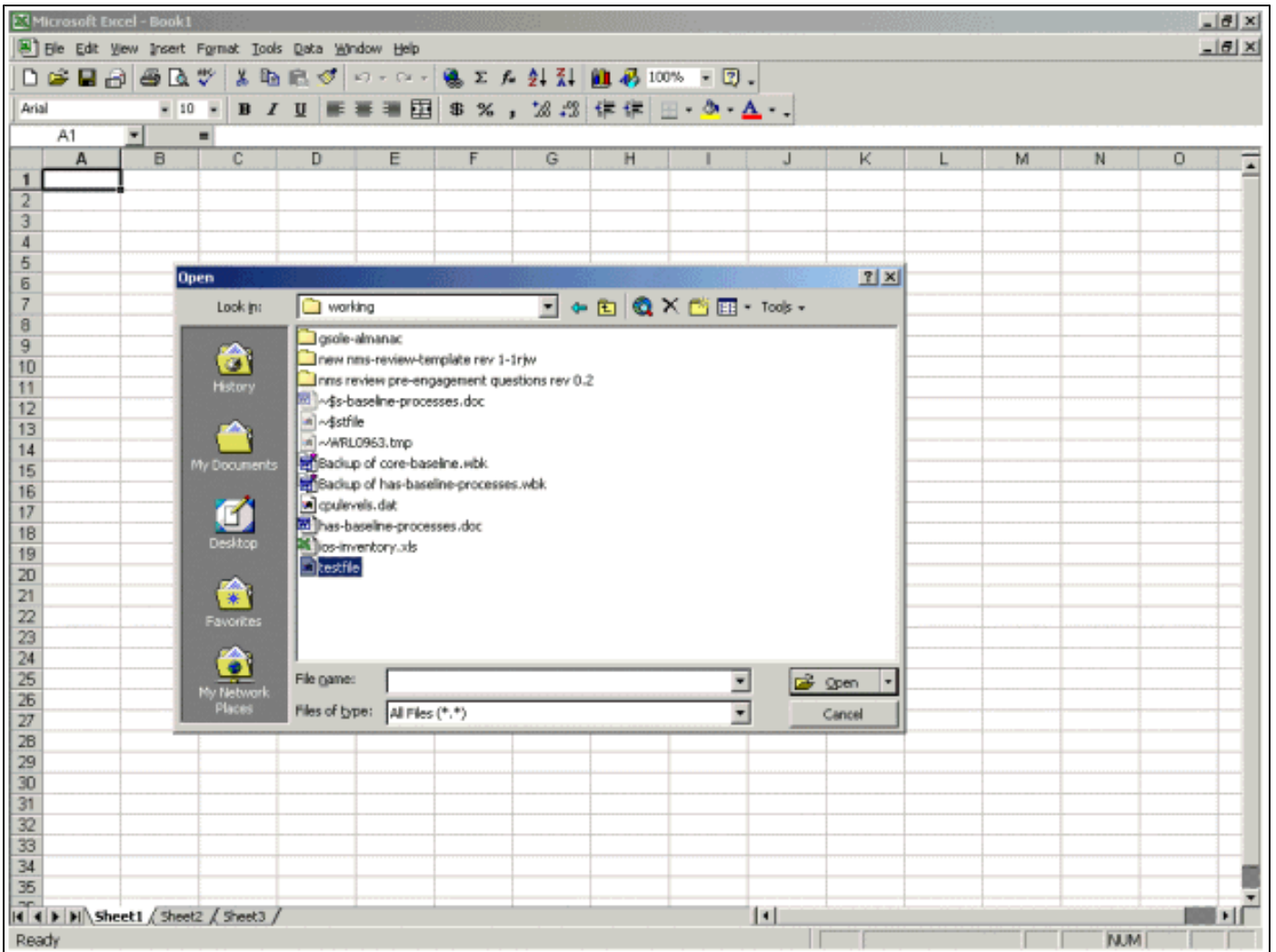
테스트 파일 출력이 UNIX 스테이션에 있으면 FTP(File Transfer Protocol)를 사용하여 PC에 전송할 수 있습니다.

자신의 스크립트를 사용하여 데이터를 수집할 수도 있습니다. 이렇게 하려면 5분마다 CPU OID에 대한 snmpget을 수행하고 결과를 .csv 파일로 덤프합니다.

4단계: 데이터 분석을 통해 임계값 결정

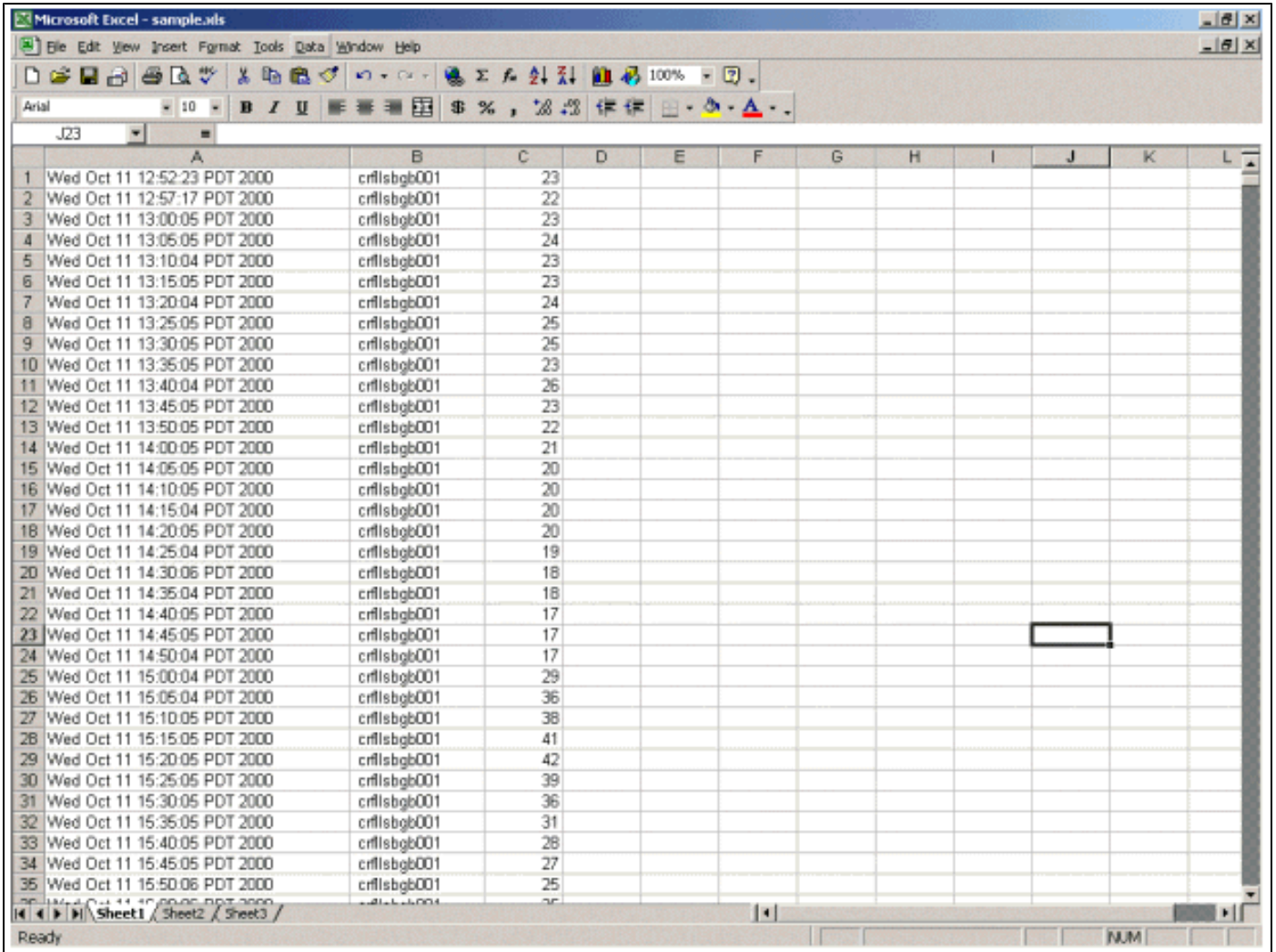
이제 데이터가 있으므로 분석을 시작할 수 있습니다. 기존 요소의 이 단계에서는 성능 또는 결함을 정확하게 측정하는 데 사용할 수 있는 임계값 설정을 결정하며, 임계값 모니터링을 켜 때 너무 많은 경보를 설정하지 않습니다. 이 작업을 수행하는 가장 쉬운 방법 중 하나는 데이터를 Microsoft Excel과 같은 스프레드시트로 가져와서 분산형 차트를 그리는 것입니다. 이 방법을 사용하면 특정 임계값을 모니터링하는 경우 특정 디바이스에서 예외 알림을 생성한 횟수를 매우 쉽게 확인할 수 있습니다. 선택한 임계값을 초과한 디바이스에서 경고 스톱이 발생할 수 있으므로 기준선을 적용하지 않고 임계값을 설정하는 것은 바람직하지 않습니다.

테스트 파일을 Excel 스프레드시트로 가져오려면 Excel을 열고 파일 > 열기를 선택한 후 데이터 파일을 선택합니다.



그런 다음 Excel 응용 프로그램에서 파일 가져오기에 대한 프롬프트를 표시합니다.

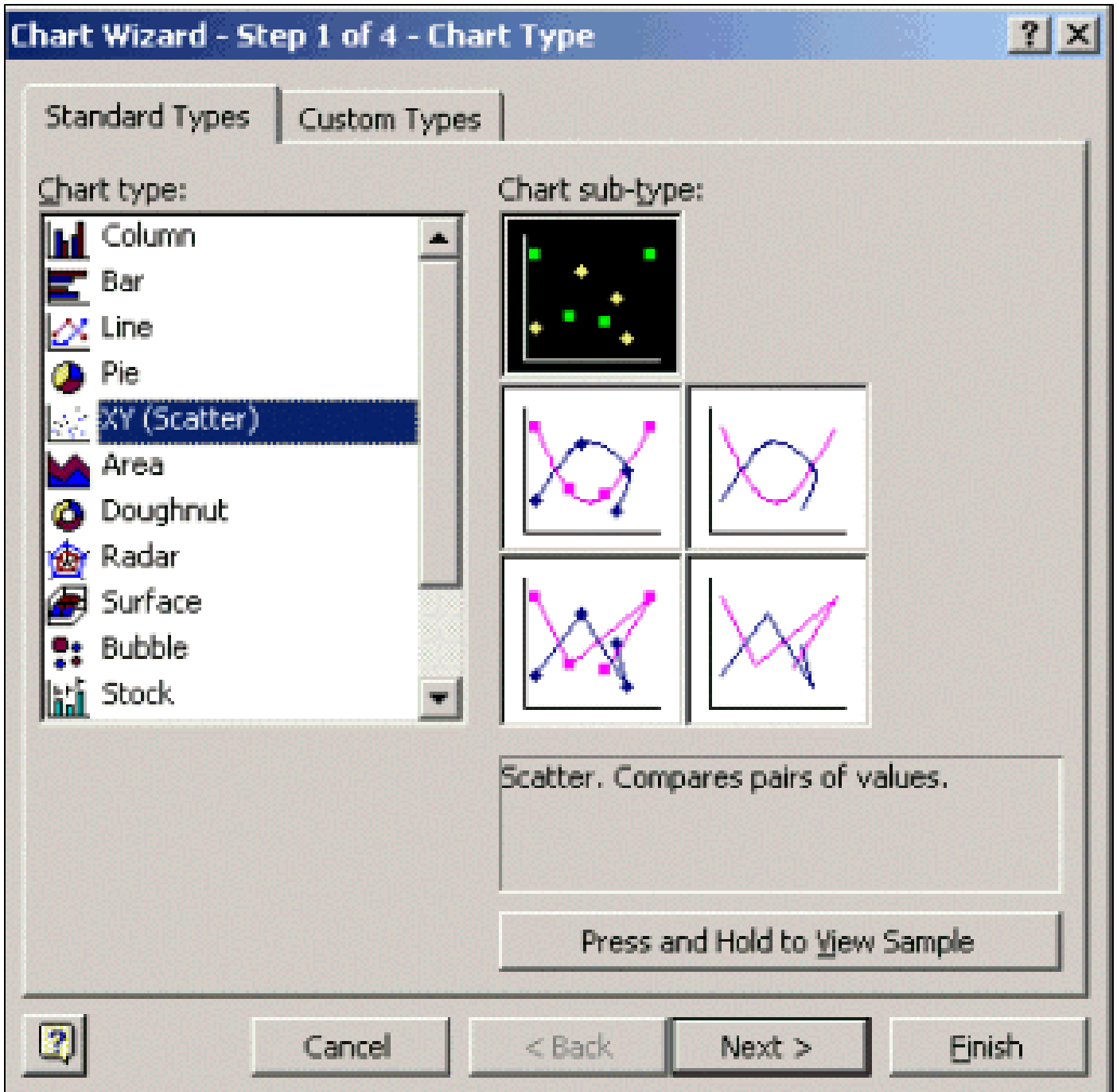
완료되면 가져온 파일은 다음 화면과 비슷하게 표시됩니다.



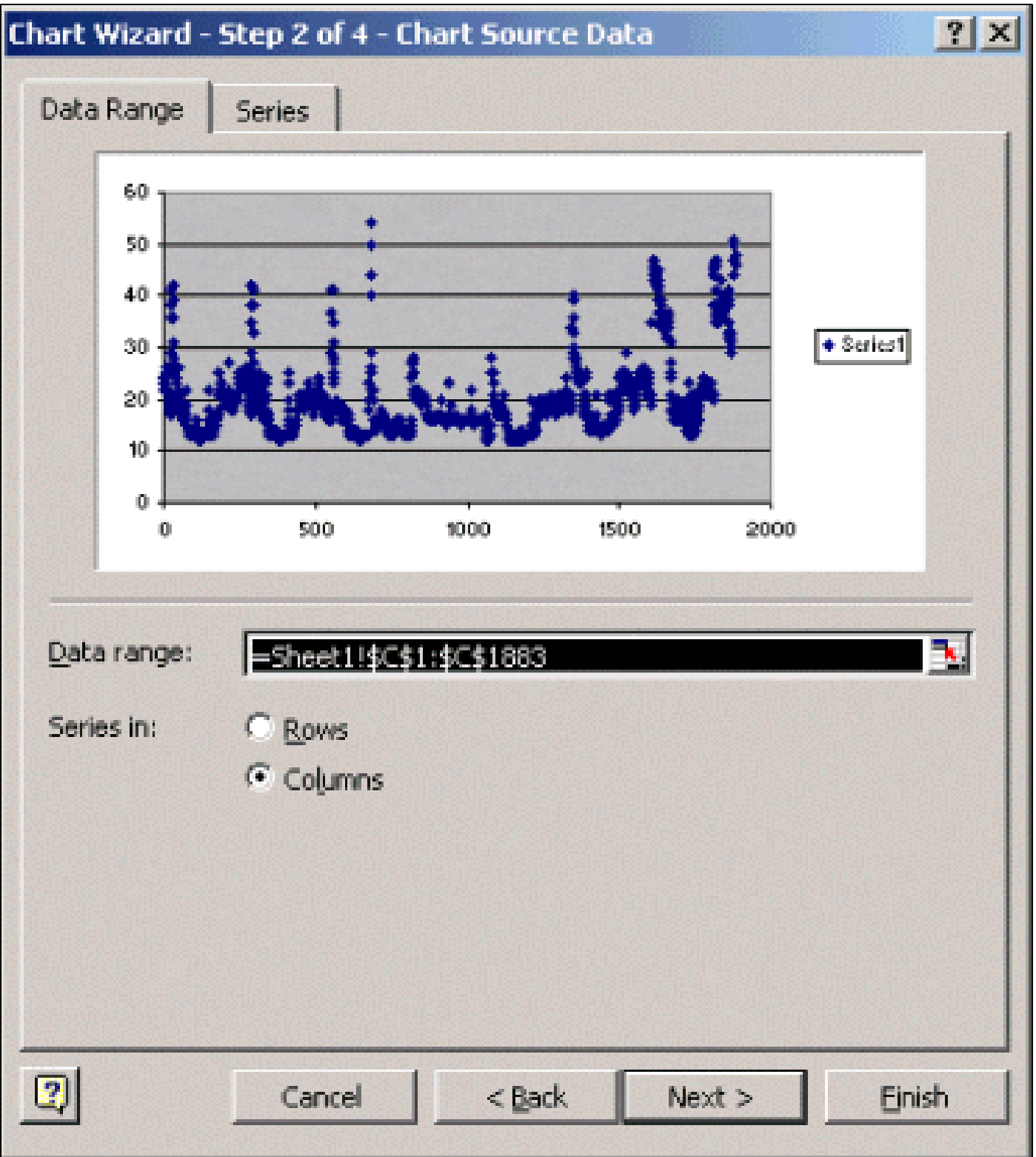
분산형 차트를 사용하면 네트워크에서 다양한 임계값 설정이 작동하는 방식을 보다 쉽게 시각화할 수 있습니다.

분산형 차트를 만들려면 가져온 파일에서 열 C를 강조 표시한 다음 차트 마법사 아이콘을 클릭합니다. 그런 다음 차트 마법사의 단계에 따라 분산형 차트를 만듭니다.

아래 표시된 것처럼 차트 마법사 1단계에서 표준 유형 탭을 선택하고 분산형 차트 유형을 선택합니다. 그런 다음 Next(다음)를 클릭합니다.



아래 표시된 것처럼 차트 마법사 2단계에서 데이터 범위 탭을 선택하고 데이터 범위 및 열 옵션을 선택합니다. Next(다음)를 클릭합니다.



아래 표시된 것처럼 차트 마법사 3단계에서 차트 제목과 X 및 Y축 값을 입력하고 다음을 누릅니다.

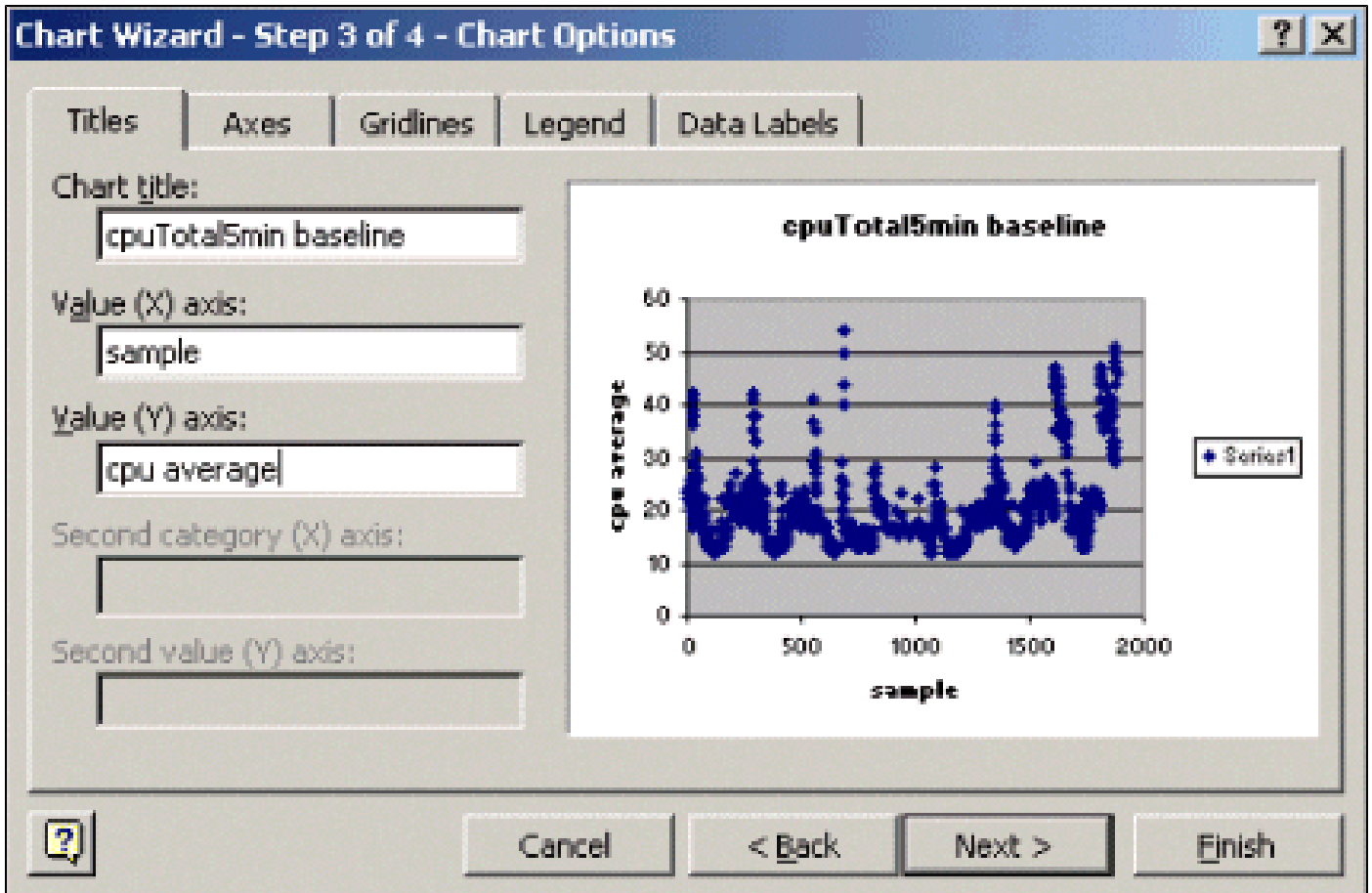


차트 마법사 4단계에서 분산형 차트를 새 페이지에 배치할지 기존 페이지에 있는 객체로 배치할지 선택합니다.

원하는 위치에 차트를 배치하려면 마침을 클릭합니다.

"만약?" 분석

이제 분석에 분산형 차트를 사용할 수 있습니다. 그러나 계속하기 전에 다음 질문을 해야 합니다.

- 공급업체(이 예에서는 공급업체가 Cisco임)가 이 MIB 변수의 임계값으로 권장하는 것은 무엇입니까?

일반적으로 코어 라우터는 평균 CPU 사용률 60%를 초과하지 않는 것이 좋습니다. 60%는 라우터에 문제가 발생하거나 네트워크에 장애가 발생할 경우 약간의 오버헤드가 필요하기 때문에 선택되었습니다. Cisco에서는 라우팅 프로토콜이 다시 계산하거나 재통합해야 할 경우 코어 라우터에 약 40%의 CPU 오버헤드가 필요할 것으로 예상합니다. 이러한 비율은 사용하는 프로토콜과 네트워크의 토폴로지 및 안정성에 따라 달라집니다.

- 임계값 설정으로 60%를 사용하는 경우 어떻게 됩니까?

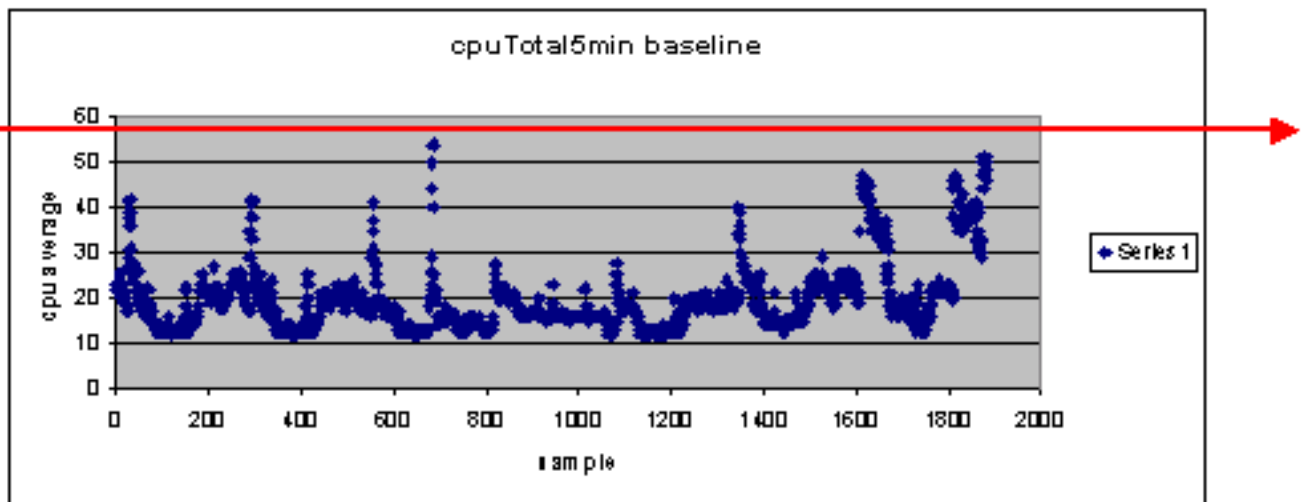
60으로 분산형 차트에 가로로 선을 그으면 데이터 포인트 중 60% CPU 사용률을 초과하는 데이터 포인트가 없습니다. 따라서 NMS(Network Management System) 스테이션에 설정된 임계값 60은 폴링 기간 동안 임계값 경보를 설정하지 않습니다. 이 라우터에는 60의 백분율을 사용할 수 있습니다. 그러나 분산형 차트에서 일부 데이터 포인트는 60에 가깝습니다. 라우터가 60% 임계값에 가까워지면 CPU가 60%에 가까워지고 있음을 미리 파악하고 해당 시점에

도달하면 어떻게 할지 계획을 세울 수 있습니다.

- 임계값을 50%로 설정하면 어떻게 합니까?

이 라우터는 이 폴링 주기 동안 네 번 사용률이 50%에 도달했으며 매번 임계값 경보를 생성했을 것으로 추정됩니다. 이 프로세스는 다른 임계값 설정이 수행할 작업을 라우터 그룹에서 확인할 때 더욱 중요합니다. 예를 들어, "전체 코어 네트워크에 대해 임계값을 50%로 설정하면 어떻게 합니까?" 보시다시피 한 가지 숫자만 선택하기는 매우 어렵습니다.

CPU 임계값 "What If" 분석



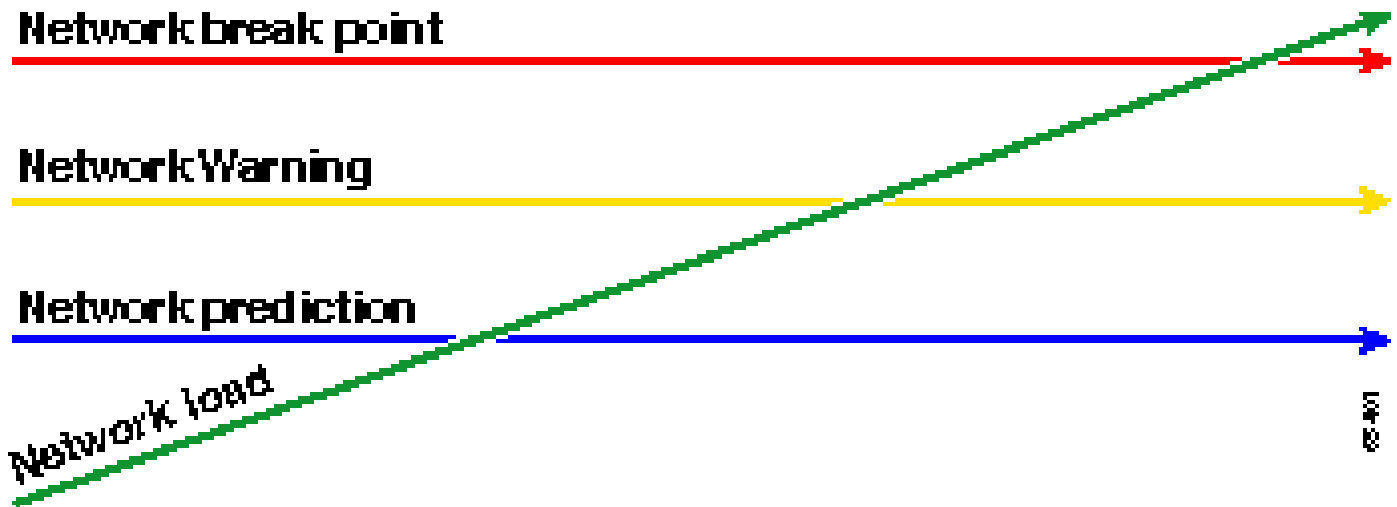
이를 더욱 쉽게 하기 위해 사용할 수 있는 한 가지 전략은 Ready, Set, Go 임계값 방법론입니다. 이 방법론은 세 개의 임계값을 연속적으로 사용합니다.

- Ready(준비) - 향후 주의해야 할 장비에 대한 예측자로 설정한 임계값
- Set(설정) - 복구, 재구성 또는 업그레이드 계획을 시작하도록 경고하는 조기 표시기로 사용되는 임계값입니다
- Go(이동) - 귀하 및/또는 공급업체가 결함 상태라고 믿고 이를 복구하기 위해 몇 가지 조치가 필요한 임계값입니다. 이 예에서는 60%입니다

다음 표는 준비, 설정, 이동 전략의 전략을 보여줍니다.

| 임계값 | 작업 | 결과 |
|-----|----------|---------------------------------------|
| 45% | 추가 조사 | 실행 계획에 대한 옵션 목록 |
| 50% | 실행 계획 수립 | 실행 계획의 단계 목록 |
| 60% | 실행 계획 구현 | 라우터가 더 이상 임계값을 초과하지 않습니다. 준비 모드로 돌아가기 |

준비됨, 설정됨, 이동 방법론은 앞서 설명한 원래 기준 차트를 변경합니다. 다음 다이어그램은 변경된 기준 요소 차트를 보여줍니다. 차트에서 다른 교차점을 식별할 수 있는 경우, 이제 이전보다 더 많은 시간을 계획하고 대응할 수 있습니다.



이 프로세스에서는 네트워크의 예외에 관심이 집중되며 다른 장치에는 관심이 없습니다. 디바이스가 임계값 미만이면 정상 상태인 것으로 가정합니다.

이러한 단계를 처음부터 고려한다면 네트워크를 안전하게 유지할 수 있도록 잘 대비할 수 있습니다. 이러한 유형의 계획을 수행하는 것은 예산 계획에도 매우 유용합니다. 상위 5개의 go 라우터, 중간 세트 라우터 및 하위 준비 라우터를 알고 있다면 라우터 종류 및 실행 계획 옵션에 따라 업그레이드에 필요한 예산을 쉽게 계획할 수 있습니다. 동일한 전략을 WAN(Wide Area Network) 링크 또는 다른 MIB OID에 사용할 수 있습니다.

5단계: 확인된 즉각적인 문제 해결

이는 베이스라인 프로세스에서 더 쉬운 부분 중 하나입니다. go 임계값 범위를 초과하는 디바이스를 식별한 후에는 해당 디바이스를 임계값 아래로 되돌리기 위한 작업 계획을 세워야 합니다.

Cisco의 TAC(Technical Assistance Center)에서 케이스를 열거나 시스템 엔지니어에게 사용 가능한 옵션에 대해 문의할 수 있습니다. 물건이 문턱에 다다르면 돈이 든다고 생각해서는 안 된다. 일부 CPU 문제는 모든 프로세스가 가장 효율적인 방식으로 실행되도록 컨피그레이션을 변경하여 해결할 수 있습니다. 예를 들어, 일부 ACL(Access Control List)은 패킷이 라우터를 통과하는 경로로 인해 라우터 CPU가 매우 높게 실행되도록 할 수 있습니다. 경우에 따라 NetFlow 스위칭을 구현하여 패킷 스위칭 경로를 변경하고 ACL이 CPU에 미치는 영향을 줄일 수 있습니다. 어떤 문제가 발생하든 이 단계에서 모든 라우터를 다시 임계값 아래로 가져와야 합니다. 그러면 나중에 너무 많은 임계값 경보가 NMS 스테이션에 범람할 위험 없이 임계값을 구현할 수 있습니다.

6단계: 테스트 임계값 모니터링

이 단계에서는 프로덕션 네트워크에서 사용할 도구를 사용하여 Lab의 임계값을 테스트합니다. 임계값을 모니터링하는 방법에는 두 가지가 있습니다. 네트워크에 가장 적합한 방법을 결정해야 합니다.

- SNMP 플랫폼 또는 기타 SNMP 모니터링 툴을 사용하여 폴링 및 비교 방법

이 방법은 트래픽을 폴링하기 위해 더 많은 네트워크 대역폭을 사용하며 SNMP 플랫폼의 처리 주기를 사용합니다.

- 임계값을 초과할 때만 알림을 전송하도록 라우터에서 RMON(Remote Monitoring) 경고 및 이벤트 컨피그레이션을 사용합니다

이 방법을 사용하면 네트워크 대역폭 사용량을 줄일 수 있을 뿐 아니라 라우터의 메모리 및 CPU 사용률도 향상됩니다.

SNMP를 사용하여 임계값 구현

HP OpenView NNM을 사용하여 SNMP 방법을 설정하려면 초기 폴링을 설정할 때와 마찬가지로 옵션 > 데이터 수집 및 임계값을 선택합니다. 그러나 이번에는 수집 메뉴에서 Store(저장), No Thresholds(임계값 없음) 대신 Store(저장), Check Thresholds(임계값 확인)를 선택합니다. 임계값을 설정한 후 여러 ping 및/또는 여러 SNMP 걸음을 전송하여 라우터의 CPU 사용률을 높일 수 있습니다. CPU가 임계값을 넘길 수 있을 만큼 높도록 강제할 수 없으면 임계값을 낮춰야 할 수도 있습니다. 어떤 경우에도 임계값 메커니즘이 작동하는지 확인해야 합니다.

이 방법을 사용할 때의 제한 사항 중 하나는 여러 임계값을 동시에 구현할 수 없다는 것입니다. 세 개의 서로 다른 동시 임계값을 설정하려면 세 개의 SNMP 플랫폼이 필요합니다. Concord [Network Health](#) 및 [Trinagy TREND](#)와 같은 [툴](#)을 사용하면 동일한 OID 인스턴스에 대해 여러 임계값을 사용할 수 있습니다.

시스템이 한 번에 하나의 임계값만 처리할 수 있는 경우 Ready(준비), Set(설정), Go(이동) 전략을 직렬 방식으로 고려할 수 있습니다. 즉, 준비 임계값에 계속 도달하면 조사를 시작하고 임계값을 해당 디바이스에 대해 설정된 수준으로 높입니다. 설정된 수준에 계속 도달하면 작업 계획을 수립하기 시작하고 해당 장치에 대한 실행 수준으로 임계값을 올립니다. 그런 다음 이동 임계값에 계속 도달하면 실행 계획을 구현합니다. 이는 세 가지 동시 임계값 방법만큼 잘 작동해야 합니다. SNMP 플랫폼 임계값 설정을 변경하는 데 시간이 조금 더 걸립니다.

RMON 경고 및 이벤트를 사용하여 임계값 구현

RMON 경고 및 이벤트 컨피그레이션을 사용하여 여러 임계값에 대해 라우터 자체 모니터링을 수행하도록 할 수 있습니다. 라우터가 과다 임계값 조건을 탐지하면 SNMP 플랫폼으로 SNMP 트랩을 전송합니다. 트랩을 전달하려면 라우터 컨피그레이션에 SNMP 트랩 수신기가 설정되어 있어야 합니다. 경고와 이벤트 사이에는 상관관계가 있습니다. 경고는 OID에서 지정된 임계값을 확인합니다. 임계값에 도달하면 경고 프로세스는 SNMP 트랩 메시지를 보내거나 RMON 로그 항목을 생성하거나 둘 다 수행할 수 있는 이벤트 프로세스를 실행합니다. 이 명령에 대한 자세한 내용은 [RMON Alarm 및 Event Configuration 명령을 참조하십시오](#).

다음 라우터 컨피그레이션 명령에는 300초마다 라우터 모니터 cpmCPUTotal5min이 있습니다. CPU가 60%를 초과하면 이벤트 1이 발생하며, CPU가 40%로 떨어지면 이벤트 2가 발생합니다. 두 경우 모두 SNMP 트랩 메시지가 커뮤니티 전용 문자열과 함께 NMS 스테이션에 전송됩니다.

Ready, Set, Go 메서드를 사용하려면 다음 컨피그레이션 문을 모두 사용합니다.

```
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
```

다음 예는 위의 명령문으로 구성된 show rmon alarm 명령의 출력을 보여줍니다.

```
<#root>
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
```

다음 예에서는 show rmon event 명령의 출력을 보여줍니다.

```
<#root>
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
```

이 두 가지 방법을 모두 사용하여 사용자 환경에 가장 적합한 방법을 확인할 수 있습니다. 당신은 심지어 방법의 조합이 잘 작동한다는 것을 발견할지도 모른다. 어떤 경우에도 모든 것이 제대로 작동하는지 확인하기 위해 랩 환경에서 테스트를 수행해야 합니다. Lab에서 테스트한 후 소규모 라우터 그룹에 제한적으로 구축하면 Operations Center에 경고를 전송하는 프로세스를 테스트할 수 있습니다.

이 경우 프로세스를 테스트하려면 임계값을 낮춰야 합니다. 생산 라우터에서 CPU를 인위적으로 올리는 것은 권장되지 않습니다. 또한 알림이 Operations Center의 NMS 스테이션에 전송될 때, 디바이스가 임계값을 초과할 때 이를 알리는 에스컬레이션 정책이 있는지 확인해야 합니다. 이러한 컨피그레이션은 Cisco IOS 버전 12.1(7)을 사용하는 Lab에서 테스트되었습니다. 문제가 발생하면 Cisco Engineering 또는 Systems Engineers에게 문의하여 IOS 버전에 버그가 있는지 확인해야 합니다.

7단계: SNMP 또는 RMON을 사용하여 임계값 모니터링 구현

Lab에서 임계값 모니터링을 철저히 테스트하고 제한된 구축 환경에서 코어 네트워크에 임계값을 구현할 준비가 된 경우. 이제 버퍼, 여유 메모리, CRC(Cyclic Redundancy Check) 오류, AMT 셀 손실 등과 같은 네트워크의 다른 중요한 MIB 변수에 대해 이 베이스라인 프로세스를 체계적으로 진행할 수 있습니다.

RMON 경고 및 이벤트 컨피그레이션을 사용하는 경우 이제 NMS 스테이션에서 폴링을 중지할 수 있습니다. 이렇게 하면 NMS 서버의 로드가 줄어들고 네트워크의 폴링 데이터 양이 줄어듭니다. 중요한 네트워크 상태 지표에 대해 체계적으로 이 프로세스를 거침으로써 네트워크 장비가 RMON Alarm and Event를 사용하여 자신을 모니터링하고 있다는 점을 쉽게 이해할 수 있습니다.

추가 MIB

이 프로세스를 학습한 후에는 다른 MIB를 조사하여 베이스라인 및 모니터링을 수행할 수 있습니다.

다음 하위 섹션에서는 일부 OID의 간략한 목록과 유용한 정보를 제공합니다.

라우터 MIB

메모리 특성은 라우터의 상태를 확인하는 데 매우 유용합니다. 정상 라우터에는 거의 항상 사용 가능한 버퍼 공간이 있어야 합니다. 라우터의 버퍼 공간이 부족해지면 CPU는 더 열심히 새 버퍼를 생성하고 수신 및 발신 패킷에 대한 버퍼를 찾아야 합니다. 버퍼에 대한 심층적인 논의는 이 문서의 범위를 벗어납니다. 그러나 일반적인 규칙으로서, 정상 라우터에는 버퍼 누락이 매우 적어야 하며 버퍼 실패 또는 사용 가능한 메모리 조건이 없어야 합니다.

| 객체 | 설명 | OID |
|----------------------------|-------------------------------------|----------------------------|
| ciscoMemoryPoolFree | 관리되는 디바이스에서 현재 사용되지 않는 메모리 풀의 바이트 수 | 1.3.6.1.4.1.9.9.48.1.1.1.6 |
| ciscoMemoryPoolLargestFree | 현재 사용되지 않 | 1.3.6.1.4.1.9.9.48.1.1.1.7 |

| | | |
|----------|--------------------------|----------------------|
| | 는 메모리 풀에서 가장 많은 연속 바이트 수 | |
| 버퍼EIMiss | 버퍼요소 누락 수입니다. | 1.3.6.1.4.1.9.2.1.12 |
| 버퍼실패 | 버퍼 할당 실패 수 | 1.3.6.1.4.1.9.2.1.46 |
| 버퍼메모리 없음 | 사용 가능한 메모리가 없기 | 1.3.6.1.4.1.9.2.1.47 |

| | | |
|--|---|--|
| | 때 문 에 실 패 한 버 퍼 수 | |
|--|---|--|

Catalyst 스위치 MIB

| 객체 | 설명 | OID |
|-----------------|---|-----------------------------|
| cpmCPUTotal5min | 최근 5분 동안의 전체 CPU 사용률. 이 개체는 OLD-CISCO-SYSTEM-MIB에서 avgBusy5 개체를 사용하지 않습니다 | 1.3.6.1.4.1.9.9.109.1.1.1.5 |
| cpmCPUTotal5sec | 지난 5초 동안의 전체 CPU 사용률. 이 개체는 이전 CISCO-SYSTEM-MIB의 busyPer 개체를 무시합니다 | 1.3.6.1.4.1.9.9.109.1.1.1.3 |
| sys트래픽 | 이전 폴링 간격의 대역폭 사용률 | 1.3.6.1.4.1.9.5.1.1.8 |
| sysTrafficPeak | 포트 카운터가 마지막으로 지 | 1.3.6.1.4.1.9.5.1.1.19 |

| | | |
|---------------------|--|-----------------------------|
| | 워졌거나 시스템이 시작된 이 후의 최대 트래픽 측 정기 값 | |
| sysTrafficPeakttime | 최대 트래 픽 측정기 값이 발생 한 이후의 시간 (100분의 1 초) | 1.3.6.1.4.1.9.5.1.1.20 |
| 포트톱사용률 | 시스템의 포트 사용 률 | 1.3.6.1.4.1.9.5.1.20.2.1.4 |
| 포트 상위 NB버퍼 오버플로 | 시스템에 있는 포트 의 버퍼 오버플로 수 | 1.3.6.1.4.1.9.5.1.20.2.1.10 |

직렬 링크 MIB

| 객체 | 설명 | OID |
|-----------------------|--|--------------------------|
| loclfInputQueueDrops | 입력 대 기열이 꽉 차서 삭제된 패킷 수 입니다. | 1.3.6.1.4.1.9.2.2.1.1.26 |
| loclfOutputQueueDrops | 출력 대 기열이 꽉 찼기 때문에 삭제된 패킷 수 입니다 | 1.3.6.1.4.1.9.2.2.1.1.27 |
| loclfCRC | 순환 중 복 체크 섬 오류 가 있는 입력 패 킷 수 | 1.3.6.1.4.1.9.2.2.1.1.12 |

RMON Alarm 및 Event Configuration 명령

경보

RMON 경보는 다음 구문으로 구성할 수 있습니다.

<#root>

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

| 요 소 | 설명 |
|-----------------------|---|
| 수 | RMON MIB의 alarmTable에 있는 alarmIndex와 동일한 경보 번호. |
| 변 하 기 쉬 우 | 모니터링할 MIB 객체. 이는 RMON MIB의 alarmTable에 사용되는 alarmVariable로 변환됩니다. |
| 간 격 | 경보가 RMON MIB의 alarmTable에 사용된 alarmInterval과 동일한 MIB 변수를 모니터링하는 시간(초)입니다. |
| 델 타 | RMON MIB의 alarmTable에 있는 alarmSampleType에 영향을 주는 MIB 변수 간의 변경을 테스트합니다. |
| 절 대 | 각 MIB 변수를 직접 테스트하며, 이는 RMON MIB의 alarmTable에 있는 alarmSampleType에 영향을 줍니다. |
| 상 승 임 계 값 | 경보가 트리거되는 값입니다. |
| 이 벤 트 번 호 | (선택 사항) 상승 또는 하강 임계값이 제한을 초과할 때 트리거할 이벤트 번호입니다. 이 값은 RMON MIB의 alarmTable에 있는 alarmRisingEventIndex 또는 alarmFallingEventIndex와 동일합니다. |
| 하 강 | 경보가 재설정되는 값입니다. |

| | |
|---------|--|
| 임계값 | |
| 소유자 문자열 | (선택 사항) 경보의 소유자를 지정합니다. 이는 RMON MIB의 alarmTable에 있는 alarmOwner와 동일합니다. |

이벤트

RMON 이벤트는 다음 구문으로 구성할 수 있습니다.

<#root>

```

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

```

| 요소 | 설명 |
|---------|---|
| 수 | 할당된 이벤트 번호. RMON MIB의 eventTable에 있는 eventIndex와 동일합니다. |
| 통나무 | (선택 사항) 이벤트가 트리거될 때 RMON 로그 항목을 생성하고 RMON MIB의 eventType을 log 또는 log-and-trap으로 설정합니다. |
| 트랩 커뮤니티 | (선택 사항) 이 트랩에 사용되는 SNMP 커뮤니티 문자열입니다. 이 행에 대한 RMON MIB의 eventType 설정을 snmp-trap 또는 log-and-trap으로 구성합니다. 이 값은 RMON MIB의 eventTable에 있는 eventCommunityValue와 동일합니다. |
| 설명 문자열 | (선택 사항) 이벤트에 대한 설명을 지정합니다. 이는 RMON MIB의 eventTable에 있는 이벤트 설명과 동일합니다. |
| 소유자 문자열 | (선택 사항) 이 이벤트의 소유자로, RMON MIB의 eventTable에 있는 eventOwner와 동일합니다. |

RMON 경고 및 이벤트 구현

RMON 경고 및 이벤트 구현에 대한 자세한 내용은 Network Management Systems Best Practices 백서의 RMON [경고 및 이벤트](#) 구현 섹션을 참조하십시오.

관련 정보

- [기술 지원 및 문서 - Cisco Systems](#)

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.