Konfigurieren von RMON-Alarm- und Ereigniseinstellungen mithilfe von SNMP-Befehlen

Inhalt

Einführung

Dieses Dokument enthält eine Beispielkonfiguration für die RMON-Alarm- und Ereigniseinstellungen (Remote Monitoring) mithilfe von SNMP-Befehlen.

Voraussetzungen

Anforderungen

Für dieses Dokument bestehen keine speziellen Anforderungen.

Verwendete Komponenten

Um die in diesem Dokument beschriebenen Verfahren zu befolgen, muss Ihr Gerät die RMON-MIB unterstützen. Sie können dies unter Cisco IOS MIB Tools überprüfen (nur registrierte Kunden).

Die Informationen in diesem Dokument wurden von den Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen.

Konventionen

Weitere Informationen zu Dokumentkonventionen finden Sie unter Cisco Technical Tips Conventions.

Hintergrundinformationen

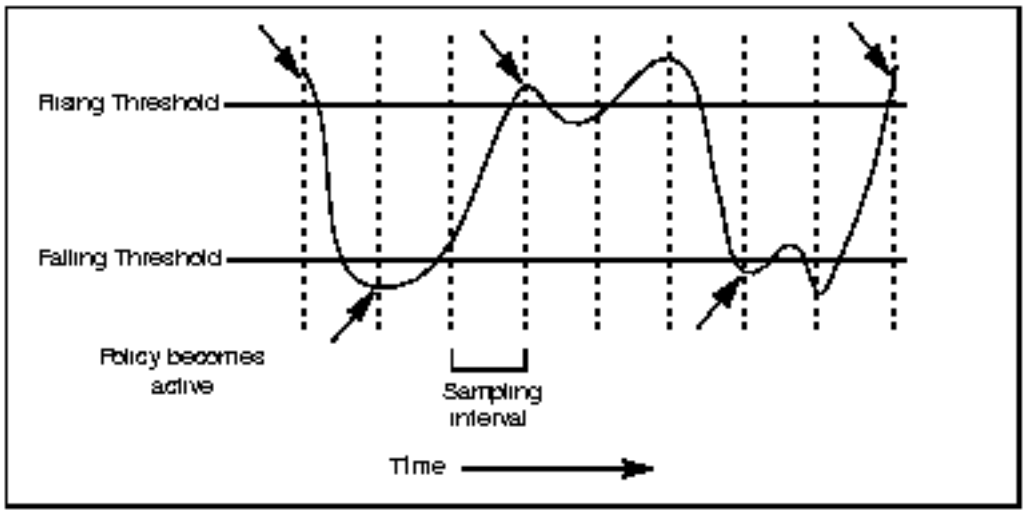
Der Zweck von RMON-Alarmen (Remote Monitoring) und -Ereignissen besteht in der Überwachung eines bestimmten MIB-Objekts auf dem Gerät und der Warnung des Systemadministrators, wenn einer dieser Werte aus dem festgelegten Bereich entfernt wird.

Der Alarm überwacht ein bestimmtes Objekt in der MIB und löst ein Ereignis aus, wenn die Bedingung (fallender oder steigender Grenzwert) erreicht wird.

Das Ereignis ist das Trap oder Protokoll, das beim Auslösen des Alarms generiert wird. Ein Beispiel für einen steigenden und fallenden Grenzwert sind:

`n=value monitored by the alarm. The falling threshold is "5" and the rising threshold is "15"`

Der Alarm löst ein Ereignis aus, wenn einer der beiden Werte erreicht wird.



Wert	Trap	Kommentar
n1 = 16	steigend	den steigenden Wert erreicht hat: 15
n2 = 4	herabfallen	den fallenden Wert erreicht hat: 5
n3 = 6	Keine	zwischen 5 und 15
n4 = 6	steigend	den steigenden Wert erreicht hat: 15
n5 = 13	Keine	guter Zustand
n6 = 20	Keine	über 15, aber nicht unter 5 seit der letzten Veranstaltung
n7 =	herabfallen	den fallenden Wert erreicht hat: 5

4		
n8 = 20	Keine	unter 5, ist seit der letzten Veranstaltung jedoch nicht über 15 gegangen.
n9 = 16	steigend	den steigenden Wert erreicht hat: 15

Sie können RMON-Alarme und -Ereignisse über die Befehlszeilenschnittstelle (CLI) auf Routern konfigurieren (siehe Konfigurieren von RMON-Alarm- und Ereigniseinstellungen über die Befehlszeilenschnittstelle) und auf Routern und Switches mithilfe von SNMP-Befehlen (Simple Network Management Protocol). Die zu ändernden Parameter sind Teil der RMON-MIB.

Schrittweise Vorgehensweise

Erstellen eines Ereignisses

Verwenden Sie diesen Befehl, um ein Ereignis zu erstellen:

```
# snmpset -c <read_write_community> <device_name> .1.3.6.1.2.1.16.9.1.1.x.y <variable type>
<value>
```

Wählen Sie zuerst die Ereignis-ID (Variable *y*) aus.

Befolgen Sie diese Prozedur, um ein Ereignis zu erstellen. Für jeden Schritt gibt es eine Beschreibung des Schritts, den Namen des zu ändernden MIB-Objekts, die Objekt-ID (OID), den <*Variablentyp*> und den <*Wert*> des Befehls Generic.

1. Löschen Sie ein altes Ereignis, das ID="y" verwendet hätte (vergewissern Sie sich, dass Sie es nicht mehr benötigen. Andernfalls verwenden Sie eine andere ID).
   ```
   * eventStatus / .1.3.6.1.2.1.16.9.1.1.7.y
   * variable type=integer
   * value=4
   ```

   Hinweis: Verwenden Sie denselben Befehl, um das Ereignis bei Bedarf zu löschen.
2. Ereigniserstellungsmodus eingeben:
   ```
   eventStatus / .1.3.6.1.2.1.16.9.1.1.7.y
   * variable type=integer
   * value v=2
   ```

3. Geben Sie die Ereignisbeschreibung an:
   ```
   * eventDescription / .1.3.6.1.2.1.16.9.1.1.2.y
   * variable type=string (for Net-snmp) or octetsting (for Openview)
   * value = a description of the event
   ```

4. Geben Sie den gewünschten Ereignistyp an:
   ```
   * eventType / .1.3.6.1.2.1.16.9.1.1.3.y
   * variable type=integer
   * value =
   "1" => none
   "2" => log
   "3" => snmp-trap
   ```

```
"4" => log-and-trap
```

5. Geben Sie den Community-String für das Trap an:
```
 * eventCommunity / .1.3.6.1.2.1.16.9.1.1.4.y
* variable type=string (for Net-snmp) or octetsting (for Openview)
* value="<trap_community_string>"
```
6. Geben Sie den Besitzer des Ereignisses an:
```
* eventOwner / .1.3.6.1.2.1.16.9.1.1.6.y
* variable type=string (for Net-snmp) or octetsting (for Openview)
* value="<event_owner>"
```
7. Aktivieren Sie die Veranstaltung:
```
* eventStatus / .1.3.6.1.2.1.16.9.1.1.7.y
* variable type=integer
* value=1
```

Erstellen eines Alarms

Verwenden Sie diesen Befehl, um einen Alarm zu erstellen:

 # snmpset -c .1.3.6.1.2.1.16.3.1.1.x.y <*read_write_community*> <*device_name*> <*variable type*> <*value*>

1. Löschen Sie einen eventuell vorhandenen Alarmmeldungen, der ID=**y** verwendet hätte (prüfen Sie zunächst, ob Sie ihn nicht mehr benötigen. Andernfalls verwenden Sie eine andere ID):
```
* alarmStatus / .1.3.6.1.2.1.16.3.1.1.12.y
* variable type=integer
* value=4
```

2. Alarmerstellungsmodus starten:
```
* alarmStatus / .1.3.6.1.2.1.16.3.1.1.12.y
* variable type=integer
* value=2
```

3. Legen Sie das Intervall (in Sekunden) fest, über das die Daten abgetastet werden, und vergleichen Sie diese mit den Schwellenwerten für steigende und fallende Werte:
```
* alarmInterval / .1.3.6.1.2.1.16.3.1.1.2.y
* variable type=integer
* value=<n_seconds>
```
4. Geben Sie die OID an, die überwacht werden soll:
```
* alarmVariable / .1.3.6.1.2.1.16.3.1.1.3.y
* variable type=objid (for Net-snmp) or objectidentifier (for Openview)
* value=<oid_to_check>
```
5. Definieren Sie den gewünschten Beispieltyp:
```
* alarmSampleType / .1.3.6.1.2.1.16.3.1.1.4.y
* variable type=integer
* value=<rising_threshold> "1" => absoluteValue "2" => deltaValue
```
6. Geben Sie an, was einen Alarm auslöst:
```
* alarmStartupAlarm / .1.3.6.1.2.1.16.3.1.1.6.y
* variable type=integer
* value=
"1" => risingAlarm
"2" => fallingAlarm
```

"3" => risingOrFallingAlarm

7. Definieren Sie den steigenden Grenzwert:
   * alarmRisingThreshold / .1.3.6.1.2.1.16.3.1.1.7.y
   * variable type=integer
   * value=<*rising_threshold*>
8. Definieren Sie den fallenden Grenzwert:
   * alarmFallingThreshold / .1.3.6.1.2.1.16.3.1.1.8.y
   * variable type=integer
   * value=<*falling_threshold*>
9. Geben Sie die Ereignis-ID an, die beim Überschreiten des steigenden Schwellenwerts ausgelöst werden soll:
   * alarmRisingEventIndex / .1.3.6.1.2.1.16.3.1.1.9.y
   * variable type=integer
   * value=<*event_ID*>
10. Geben Sie die Ereignis-ID an, wenn der fallende Grenzwert überschritten wird:
    * alarmFallingEventIndex / .1.3.6.1.2.1.16.3.1.1.9.y
    * variable type=integer
    * value=<*event_ID*>
11. Geben Sie den Eigentümer des Alarms an:
    * alarmOwner / .1.3.6.1.2.1.16.3.1.1.11.y
    * variable type=string (for Net-snmp) or octetsting (for Openview)
    * value=<*owner*>
12. Aktivieren Sie den Alarm:
    * alarmStatus / .1.3.6.1.2.1.16.3.1.1.12.y
    * variable type=integer
    * value=1

Beispiel

In diesem Beispiel wird **Safari** zum Senden eines Traps verwendet, wenn die Anzahl der Bytes, die in den letzten zwei Minuten an Schnittstelle 12 übertragen werden, über 14000000 oder unter 10 liegt.

Safari ist die Cisco IOS 2500 Software (C2500-JS-L), Version 12.1(9), VERSION-SOFTWARE (fc1).

Dieses Beispiel wurde auch erfolgreich mit WS-C6506-Software, Version NmpSW: 6.1(1b)

Hinweis: Im Catalyst gibt es keinen CLI-Befehl zum Überprüfen der Konfiguration. Dies kann jedoch mit dem Befehl **snmpwalk** auf dem Server erfolgen.

Auf dem Router und dem Switch bleibt diese Konfiguration beim erneuten Laden erhalten.

```
safari# show rmon events
Event table is empty

# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.7.123 integer 4
16.9.1.1.7.123 = 4
# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.7.123 integer 2
16.9.1.1.7.123 = 2

safari#show rmon events
Event 123 is under creation, owned by
```

```
Description is
Event firing causes nothing, last fired 00:00:00

# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.2.123 string "test_event"
16.9.1.1.2.123 = "test_event"
# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.3.123 integer 4
16.9.1.1.3.123 = 4
# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.4.123 string "public"
16.9.1.1.4.123 = "public"
# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.6.123  string "event_owner"
16.9.1.1.6.123 = "event_owner"
# snmpset -c private safari .1.3.6.1.2.1.16.9.1.1.7.123 integer 1
16.9.1.1.7.123 = 1

safari# show rmon events
Event 123 is active, owned by event_owner
Description is test_event
Event firing  causes log and trap to community public, last fired 00:00:00

safari# show rmon alarm
Alarm table is empty

# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.12.321 integer 2
16.3.1.1.12.321 = 2

safari# show rmon alarm
Alarm 321 is under creation, owned by
Monitors ccitt.0 every 10 second(s)
Taking absolute samples, last value was 0
Rising threshold is 0, assigned to event 0
Falling threshold is 0, assigned to event 0
On startup enable rising or falling alarm

# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.2.321 integer 120
16.3.1.1.2.321 = 120
# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.3.321 objid .1.3.6.1.2.1.2.2.1.10.12
16.3.1.1.3.321 = OID: interfaces.ifTable.ifEntry.ifInOctets.12
# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.4.321 integer 2
16.3.1.1.4.321 = 2
# snmpset -c private  safari .1.3.6.1.2.1.16.3.1.1.6.321 integer 3
16.3.1.1.6.321 = 3
# snmpset -c  private safari .1.3.6.1.2.1.16.3.1.1.7.321 integer 140000000
16.3.1.1.7.321 = 140000000
# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.8.321 integer 10
16.3.1.1.8.321 = 10
# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.9.321  integer 123
16.3.1.1.9.321 = 123
# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.10.321  integer 123
16.3.1.1.10.321 = 123
# snmpset -c private safari .1.3.6.1.2.1.16.3.1.1.11.321  string "alarm_owner"
16.3.1.1.11.321 = "alarm_owner"
# snmpset -c private safari  .1.3.6.1.2.1.16.3.1.1.12.321 integer 1
16.3.1.1.12.321 = 1

safari# show rmon alarm
Alarm 321 is active, owned by alarm_owner
Monitors ifEntry.10.1 every 120 second(s)
Taking delta samples, last value was 130244
Rising threshold is 140000000, assigned to event 123
Falling threshold is 10, assigned to event 123
On startup enable rising or falling alarm
```

Überprüfen

Für diese Konfiguration ist derzeit kein Überprüfungsverfahren verfügbar.

Fehlerbehebung

Für diese Konfiguration sind derzeit keine spezifischen Informationen zur Fehlerbehebung verfügbar.

Zugehörige Informationen

- Konfigurieren von RMON-Alarm- und Ereigniseinstellungen über die Befehlszeilenschnittstelle
- Ereignis-MIB-Unterstützung
- RFC 1757
- Technischer Support - Cisco Systems