

Frequently Asked Questions About CEMF Alarms

Document ID: 24150

Questions

- Is there a quick way to delete all alarms in the system?
 - How can I check what traps I'm receiving, and how do I map them to alarms?
-

Q. Is there a quick way to delete all alarms in the system?

A. Yes, you can use the **alarmDeleter** utility in the <CEMFROOT>/bin directory. You need to configure the [**AlarmDeleter**] section in the <CEMFROOT>/config/init/alarmDeleter.ini configuration file. In this file you specify the age of the alarms to be deleted, so if you want all alarms to be deleted, set the age to **0** (that is, set all attributes in alarmDeleter.ini beginning with **age** to **0** (under section [**AlarmDeleter**])).

In addition, the alarmDeleter.ini file contains the **deleteAllAlarms** variable which can be set to **0** to clear only cleared alarms, and set to **1** to clear active and cleared alarms (all alarms). If you want all alarms to be deleted (not just cleared alarms), then you also need to set **deleteAllAlarms** to **1**.

See page 80 of **Cisco Element Management Framework Physical Architecture** for more information on alarm deletion.

Q. How can I check what traps I'm receiving, and how do I map them to alarms?

A. Follow these steps:

3. Check to see that your workstation is receiving traps

Snoop — The **snoop** program is provided with Solaris (/usr/sbin/snoop). SNMP traps are sent to UDP port 162, so the following command can be issued as root:

```
snoop port 162 hostname <address-of-device-raising-traps>
```

and results in messages like the following when traps are raised:

```
<code>snmp-device -> your-hostname UDP D=162 S=3351  
LEN=146</code>
```

2. Logging Traps Received (as of Cisco EMF Version 3.1)

If you need detailed log information on what traps have been received, turn on full logging in the trapmanager process by editing <CEMFROOT>/config/init/trapmanager.ini This file normally contains:

```
ER_ALL=0
```

change this to

```
ER_ALL=1
```

Then re-start CEMF (<CODE>**cemf stop ; cemf start**</CODE>), and you should see the log file in <CEMFROOT>/logs/trapManager.log .

1. Trap mapping files

Assuming that you have a specific **Controller** rather than a generic one, your trap mapping files should all be placed in <CEMFROOT>/config/<controller>/trapMappingFiles.

For example, if you want to have a file to deal with device specific traps, a file to deal with generic traps, and a catch-all file to ensure all traps are raised even if you're not sure exactly what type. These files are examined in order according to the Sxx prefix of their names, so first you might have a device specific file, such as **S10DeviceSpecificTrapMappingFile** then **S20GenericTrapMappingFile** and finally the catch-all **S30BucketTrapMappingFile**.

Here are some example entries for SNMP V2c traps in each file:

```
.....:
S10DeviceSpecificTrapMappingFile
.....:
#
# Mappings for a device-specific trap.
# the generic and specific IDs are set to -1 for SNMP v2c traps
#
# and the trap OID is mapped to the enterprise ID, only the
#
# enterprise ID is used in this rule for trap mapping.
#
TRAP-MAPPING {
  NAME aDeviceSpecificNotification
  ENTERPRISE
  GENERIC-ID -1
  SPECIFIC-ID -1
  SEVERITY-MAPPING { STATIC { SEVERITY major } }
  CLASS-MAPPING { STATIC { CLASS aDeviceSpecificNotification } }
}

.....:
S20GenericTrapMappingFile
.....:
#
#
# Mapping for a generic SNMP v1 "link down" trap
#
# only the generic ID is used in this rule for trap mapping
#
#
TRAP-MAPPING {
  NAME linkDown
  ENTERPRISE *
  GENERIC-ID 2
  SPECIFIC-ID -1
  SEVERITY-MAPPING { STATIC { SEVERITY major } }
  CLASS-MAPPING { STATIC { CLASS linkDown } }
}

.....:
S30BucketTrapMappingFile
.....:
#
# This rule would fire for any trap received which had
# not been mapped with any previously specified rules
#
```

```
TRAP-MAPPING {
  NAME catchall
  ENTERPRISE *
  GENERIC-ID -1
  SPECIFIC-ID -1
  SEVERITY-MAPPING { STATIC { SEVERITY major } }
  CLASS-MAPPING { STATIC { CLASS catchall } }
}
```

Remember to define your alarm classes in <CEMFROOT>/**config/objectTypes**. For example, entries might include:

```
OBJECTCLASS
  NAME linkDown
  INHERITS technologyAlarm
  KEYWORD "AlarmMessage", "Link Down!"
ENDOBJECTCLASS
```

Once you restart your controller to parse in the new trap mapping files, you may need to close and re-open the **Event Browser** in order to receive the correct, updated, trap description based on the newly-parsed trap mapping files.

Related Information

- [CEMF Knowledge Base](#)

All contents are Copyright © 1992--2002 Cisco Systems, Inc. All rights reserved. Important Notices and Privacy Statement.

Updated: Jul 31, 2007

Document ID: 24150
