



Configuring Remote Monitoring (RMON)

This chapter provides information on configuring the Remote Monitoring (RMON) features of your CSS. Information in this chapter applies to all CSS models, except where noted.

This chapter contains the following major sections:

- [RMON Overview](#)
- [RMON Configuration Considerations](#)
- [Configuring an RMON Event](#)
- [Configuring an RMON Alarm](#)
- [Configuring an RMON History](#)
- [Viewing RMON Information](#)

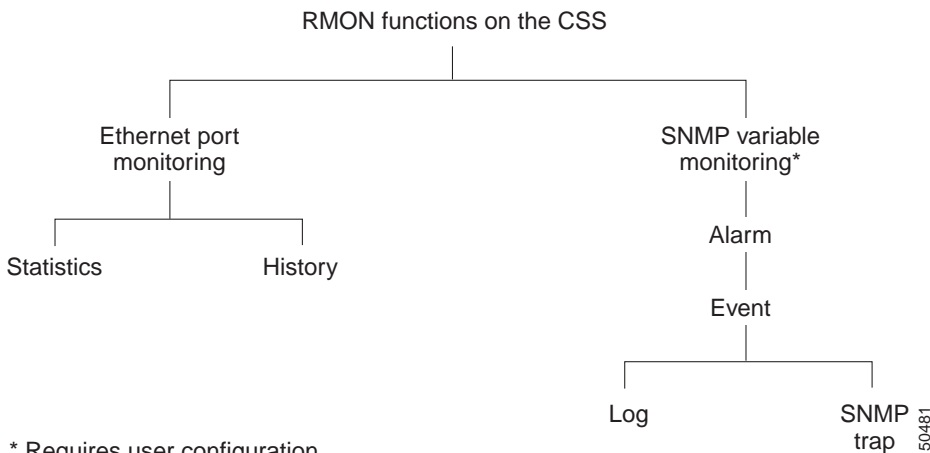
RMON Overview

RMON allows you to remotely monitor and analyze the activity of packets on CSS Ethernet ports. RMON also allows alarm configuration for monitoring MIB objects, and the event configuration to notify you of these alarm conditions. For detailed information about RMON and its MIB objects, refer to RFC 1757.

The version of RMON provided on the CSS is a subset of the RMON-1 groups (Figure 6-1). The CSS supports the following groups:

- Group 1 - (Statistics) Provides data about all Ethernet ports on a CSS. You cannot configure RMON statistics. You can only view them.
- Group 2 - (History) Provides data about the Ethernet ports over an historical period. Histories are preconfigured for each port. You can configure additional port histories.
- Group 3 - (Alarm) Allows you to create an alarm and configure the conditions, based on a MIB object, to trigger an alarm when changes are detected.
- Group 9 - (Event) Allows you to create an event and configure the event action that is to be performed when an associated alarm occurs.

Figure 6-1 Supported RMON Functions on the CSS



RMON Configuration Considerations

Consider the following points before you implement RMON functionality on your CSS:

- You can configure an RMON event, alarm, and history. You cannot configure CSS attributes for RMON statistics. Statistics for the ports are viewable only by using the **show rmon** command.
- You cannot change the configuration for an RMON history after you activate it. If you need to change the RMON history configuration after activation, you must delete it first and then recreate the RMON history with the necessary changes. You can change your RMON history configuration at any time before you activate it.
- You must assign an RMON event to an RMON alarm before the alarm can be activated. The event must exist and must be activated before it can be assigned to an RMON alarm.
- RMON histories are preconfigured for each Ethernet port. Though these histories cannot be deleted or modified, you can add history entries for a port. For more information on the preconfigured histories and adding more history entries, see the [“Configuring an RMON History”](#) section.

Configuring an RMON Event

An RMON event is the action that occurs when an associated RMON alarm is triggered. When an alarm event occurs, it can be configured to generate a log event, a trap to an SNMP network management station, or both. For information on viewing alarm events in log files, see the [“Viewing Events in a Log File”](#) section. Refer to [Chapter 5, Configuring Simple Network Management Protocol \(SNMP\)](#), for information on configuring SNMP on your CSS.

The following sections describe how to configure an RMON event.

- [RMON Event Configuration Quick Start](#)
- [Creating an Index for an RMON Event](#)
- [Deleting an RMON Event Index](#)
- [Setting the RMON Event Attributes](#)
- [Activating an RMON Event](#)
- [Suspending an RMON Alarm](#)

RMON Event Configuration Quick Start

[Table 6-1](#) provides a quick overview of the steps required to configure the attributes for an RMON event. Each step includes the CLI command required to complete the task. For a complete description of each feature and all the options associated with the CLI command, see the sections following [Table 6-1](#). For information on configuring an alarm and associating this event to an alarm, see the [“Configuring an RMON Alarm”](#) section.

Table 6-1 RMON Event Configuration Quick Start

Steps and Possible Settings

1. Create an RMON event index from global configuration mode. Enter an integer from 1 to 65534.


```
(config)# rmon-event 1
Create Event <1>, [y/n]:y
```

2. Assign an existing SNMP community for this event. The specified *community_name* is the name of an SNMP community configured using the **snmp trap-host** command. This step is required only if the traps are sent to an SNMP network management station.


```
(config-rmonevent[1])# community moonbase_alpha
```

3. Provide a description for the event. Enter a quoted text string with a maximum of 127 characters including spaces.


```
(config-rmonevent[1])# description "This event occurs when service
connections exceed 100"
```

4. Assign the owner who defined and is using the resources of the event. Enter a quoted string with a maximum of 127 characters including spaces. You must define the owner before you can activate the event.


```
(config-rmonevent[1])# owner "Boston Tech Lab"
```

5. Specify the type of event notification. The type determines where the notification is sent.


```
(config-rmonevent[1])# type log-and-trap
```

6. Activate the event.


```
(config-rmonevent[1])# active
```

The following running-configuration example shows the results of entering the commands in [Table 6-1](#).

```
!***** RMON EVENT *****
rmon-event 1
  community moonbase_alpha
  description "This event occurs when service connections exceed 100"
  owner "Boston Tech Lab"
  type log-and-trap
  active
```

Creating an Index for an RMON Event

The RMON event index identifies the event to the CSS. This index allows you to assign specific configuration attributes to the RMON event. When you create an RMON event index, you access the configuration mode for that event automatically.

Use the **rmon-event** *index* command to create an event index. You can access this command from any configuration mode except the boot and RMON alarm configuration modes. The *index* is a number from 1 to 65534.



Note

The RMON event index 65535 is administratively predefined and cannot be modified. If you enter this index number, a message similar to the following appears: %% Index internally used. Administrative control not allowed.

For example, to create an RMON event with an identifier of 1, access global configuration mode and enter:

```
(config)# rmon-event 1
```

To view a list of existing RMON event configuration identifiers, enter:

```
(config)# rmon-event ?
```

After you create the index for the event, the prompt changes to (config-rmonevent[1]). Define the event as described in the [“Setting the RMON Event Attributes”](#) section.

Modifying the Attributes for an Existing RMON Event Index

Use the **suspend** command to deactivate the RMON event and make attribute changes.

Deleting an RMON Event Index

If you have an active RMON event index that you no longer need, use the **no rmon-event** command. This command is available in the RMON alarm, RMON event, RMON history, and global configuration modes.

To delete RMON event 1 and its configuration, enter:

```
(config)# no rmon-event 1
Delete Event <1>, [y/n]:y
```

Setting the RMON Event Attributes

After you create an RMON event index, access RMON event configuration mode for the event identifier and set its attributes. This section includes the following topics:

- [Defining an Event Community](#)
- [Describing an Event](#)
- [Assigning an Owner](#)
- [Defining the Notification of an Event](#)

After you set the attributes, activate the event as described in the “[Activating an RMON Event](#)” section.

If an RMON event is activated and you want to make modifications to certain event attributes, you must first suspend the RMON event (as described in the “[Suspending an RMON Event](#)” section). Ensure the RMON event is not assigned to an RMON alarm.

Defining an Event Community

When an alarm event occurs and the event is configured to send an SNMP trap, the CSS sends the trap to the trap host with the specified community. If no community is specified the CSS automatically uses the default event community of “public”.

Use the **community** *community_name* command to define a community for an unactivated event. The *community_name* variable is the name of the SNMP community you configured using the **snmp trap-host** command (refer to [Chapter 5, Configuring Simple Network Management Protocol \(SNMP\)](#)).

To view a list of currently configured community strings (configured using the **snmp trap-host** command), enter **rmon-event community ?**

For example, to define the SNMP *moonbase_alpha* community for this event, enter:

```
(config-rmonevent [1])# community moonbase_alpha
```

To reset the community back to public, enter:

```
(config-rmonevent [1])# no community
```

Describing an Event

When an alarm event occurs, the CSS sends a description with the event notification. Because a description is not generated automatically, you must provide one. Use the **description** “*description*” command to provide a description. The *description* variable is the description for the RMON event. Enter a quoted text string with a maximum of 127 characters.

To provide a description for the event, enter:

```
(config-rmonevent [1])# description "This event occurs when service connections exceed 100"
```

To remove the description from the event, enter:

```
(config-rmonevent [1])# no description
```

Assigning an Owner

You must define the entity who configured this RMON event and is using the resources assigned to it. Use the **owner** “*owner_name*” command to define the owner. The *owner_name* variable is a quoted text string with a maximum of 127 characters. The owner for the event must be the same as the owner for the alarm.

To define the owner named Boston Tech Lab, enter:

```
(config-rmonevent [1])# owner "Boston Tech Lab"
```

To remove the owner of the RMON event, enter:

```
(config-rmonevent [1])# no owner
```

You must reassign an owner before you can reactivate the RMON event.

Defining the Notification of an Event

When an RMON event occurs, the event type determines where the CSS sends the event notification.

- A log event type designates that the event notification is made in a CSS log location (for example, CSS disk log file or session). For information on viewing log files, see the [“Viewing Events in a Log File”](#) section.

To define the event as a log type (default), enter:

```
(config-rmonevent [1])# type log
```

- A trap event type designates that a trap is sent to a SNMP network management station. To define the event as a trap type, enter:

```
(config-rmonevent [1])# type trap
```



Note When you want the event to send a trap to a network management station, you need to configure SNMP. Refer to [Chapter 5, Configuring Simple Network Management Protocol \(SNMP\)](#), for information on SNMP.

- You can also designate that the event type is both log and trap. To define the event as both log and trap types, enter:

```
(config-rmonevent [1])# type log-and-trap
```

To reset the RMON event type back to log, enter:

```
(config-rmonevent [1])# no type
```

Activating an RMON Event

After you configure the event attributes, activate the event. However, before you can activate an event, you must specify the owner of the event as described in the [“Assigning an Owner”](#) section.

To activate the event, enter:

```
(config-rmonevent [1])# active
```

Note the following when activating an RMON event, once an RMON event is activated and you want to make modifications to certain event attributes, you must first suspend the RMON event. Ensure the RMON event is not assigned to an RMON alarm.

Suspending an RMON Event

Suspending an RMON event deactivates it, allowing you to make changes to its configuration settings. Use the **suspend** command to suspend an event.

When you suspend an RMON event, ensure that the event is not assigned to an RMON alarm.

For example:

```
(config-rmonevent [1]) # suspend
```

Configuring an RMON Alarm

An RMON alarm allows you to monitor a MIB object for a desired transitory state. An alarm periodically takes samples of the object's value and compares them to the configured thresholds.

RMON allows you to configure two types of sampling, absolute and delta:

- Absolute sampling compares the sample value directly to the threshold. This sampling is similar to a gauge, recording values that go up or down.
- Delta sampling subtracts the current sample value from the last sample taken and then compares the difference to the threshold. This sampling is similar to a counter, recording a value that is constantly increasing.

When a sample value crosses an alarm threshold, an associated event is generated. To limit the number of generated events, only one event is generated when a threshold is crossed. The CSS does not generate additional events until an opposite threshold is crossed. For example, when a rising threshold is crossed, one event is generated. The next event occurs only when a falling threshold is crossed.

When you associate an event to an alarm and an alarm occurs, the event defines the action the CSS takes when an alarm occurs. For more information on events, see the [“Configuring an RMON Event”](#) section.

Figure 6-2 is an example of absolute sampling.

Figure 6-2 Example of Absolute Sampling

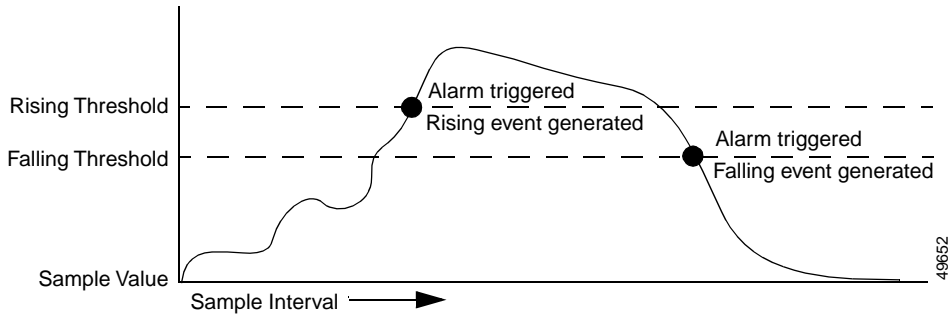
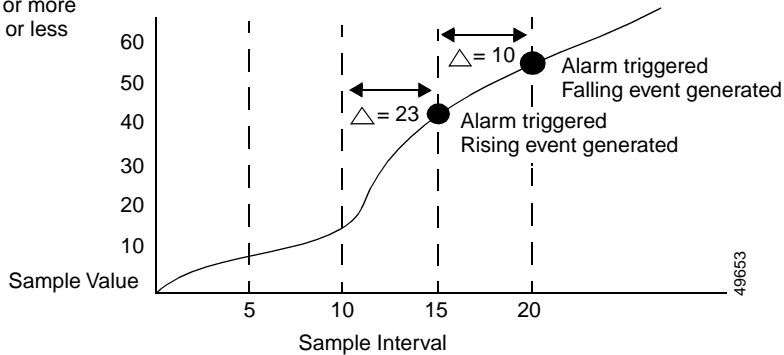


Figure 6-3 is an example of delta sampling.

Figure 6-3 Example of Delta Sampling

Rising Threshold = 20 or more
Falling Threshold = 10 or less



This section includes the following topics:

- [RMON Alarm Configuration Quick Start](#)
- [Creating an Index for an RMON Alarm](#)
- [Deleting an RMON Alarm Index](#)
- [Setting the RMON Alarm Attributes](#)

- [Activating an RMON Alarm](#)
- [Suspending an RMON Alarm](#)

RMON Alarm Configuration Quick Start

[Table 6-2](#) provides a quick overview of the steps required to configure the attributes for an RMON alarm. Each step includes the CLI command required to complete the task. For a complete description of each feature and all the options associated with the CLI command, see the sections following [Table 6-2](#).

Table 6-2 RMON Alarm Configuration Quick Start

Steps and Possible Settings

1. Create an RMON alarm index from global configuration mode. Enter an integer from 1 to 65534.


```
(config)# rmon-alarm 1
Create Alarm <1>, [y/n]:y
```
 2. Assign the owner who defined and is using the resources of the alarm. Enter a quoted string with a maximum of 127 characters including spaces. The owner must be the same as the owner for the event.


```
(config-rmonalarm[1])# owner "Boston Tech Lab"
```
 3. Define the MIB object for the sample variable. For example, for the current number of connections for this service, enter `apSvcConnections`. To see a list of objects, use the `sample-variable ?` command. For detailed information about an object, use the `lookup` command.


```
(config-rmonalarm[1])# sample-variable apSvcConnections
```
 4. Define the sampling type. The options are **absolute** or **delta** (default).


```
(config-rmonalarm[1])# sample-type absolute
```
 5. Define the startup alarm type. The options are **falling**, **rising** (default), or **rising-and-falling**.


```
(config-rmonalarm[1])# startup-type rising-and-falling
```
 6. Define the rising threshold. Enter an integer from 0 (default) to 4294967295.


```
(config-rmonalarm[1])# rising-threshold 100
```
-

Table 6-2 RMON Alarm Configuration Quick Start (continued)**Steps and Possible Settings**

7. Associate the rising event with an existing RMON event. Enter an integer from 0 to 65534. If you enter 0, no event is generated.

```
(config-rmonalarm[1])# rising-event 1
```

8. Define the falling threshold. Enter an integer from 0 (default) to 4294967295.

```
(config-rmonalarm[1])# falling-threshold 90
```

9. Associate the falling event with an existing RMON event. Enter a number from 0 to 65534. If you enter 0, no event is generated.

```
(config-rmonalarm[1])# falling-event 2
```

10. Specify the sampling interval for the RMON alarm. The interval is in seconds. Enter an integer from 1 to 65535 (default is 300).

```
(config-rmonalarm[1])# sample-interval 30
```

11. Activate the alarm.

```
(config-rmonalarm[1])# active
```

The following running-configuration example shows the results of entering the commands in [Table 6-2](#).

```
!***** RMON ALARM *****
rmon-alarm 1
  owner "Boston Tech Lab"
  sample-variable apSvcConnections.
  sample-type absolute
  startup-type rising-and-falling
  rising-threshold 100
  rising-event 1
  falling-threshold 90
  falling-event 2
  sample-interval 30
  active
```

Creating an Index for an RMON Alarm

The RMON alarm index identifies the alarm to the CSS. This index allows you to assign specific configuration attributes to the RMON alarm. When you create an RMON alarm index, you access the configuration mode for that alarm automatically.

Use the **rmon-alarm** *index* command to create an alarm index. You can access this command from any configuration mode except the boot and RMON history configuration modes. The *index* is an integer from 1 to 65534.



Note

The RMON alarm index 65535 is administratively predefined and cannot be modified. If you enter this index number, a message similar to the following appears: %% Index internally used. Administrative control not allowed.

To create an RMON alarm with an identifier of 1, access global configuration mode and enter:

```
(config)# rmon-alarm 1
```

To see a list of existing RMON alarm configuration identifiers, enter **rmon-alarm ?**.

After you create the identifier for the alarm, the prompt changes to (config-rmonalarm[1]). Now you can define the alarm as described in the [“Setting the RMON Alarm Attributes”](#) section.

Modifying the Attributes for an Existing RMON Alarm Index

Before you can modify attributes for an active RMON alarm, you must deactivate the alarm. Use the **suspend** command to deactivate the RMON alarm and make attribute changes.

For example:

```
(config-rmonalarm[1])# suspend
```

Deleting an RMON Alarm Index

If you have an active RMON alarm index that you no longer need and want to delete it, use the **no rmon-alarm** command. This command is available in the RMON alarm, RMON event, RMON history, and global configuration modes.

For example, to delete RMON alarm 1 and its configuration, enter:

```
(config)# no rmon-alarm 1
Delete Alarm <1>, [y/n]:y
```

Setting the RMON Alarm Attributes

After you create an RMON alarm index, access RMON alarm configuration mode for an existing inactive alarm identifier and set its attributes. This section includes the following topics:

- [Assigning an Owner](#)
- [Finding and Defining a Sample Variable](#)
- [Defining the Absolute or Delta Sampling Method](#)
- [Defining a Rising Threshold and Rising Event](#)
- [Defining a Falling Threshold and Index](#)
- [Defining a Startup Alarm](#)
- [Defining the Sampling Interval](#)

After you set all of the attributes, activate the alarm as described in the [“Activating an RMON Alarm”](#) section.

If an RMON alarm is activated and you want to make modifications to certain alarm attributes, you must first suspend the RMON alarm (as described in the [“Suspending an RMON Alarm”](#) section).

Assigning an Owner

Define the owner who configured the RMON alarm and is using the resources assigned to the alarm. To define the owner, use the **owner** “*owner_name*” command. You must reassign an owner before you can reactivate the RMON alarm.

The *owner_name* variable is a quoted text string with a maximum of 127 characters. Enter the same name as the owner of the event.

To define the owner named Boston Tech Lab, enter:

```
(config-rmonalarm[1])# owner "Boston Tech Lab"
```

To remove the owner of the RMON alarm, enter:

```
(config-rmonalarm[1])# no owner
```

Finding and Defining a Sample Variable

For an alarm condition, RMON samples a configured sample variable associated with a MIB object. MIB objects to consider include:

- *svcExt.mib* - Contains service objects (for example, *apSvcConnections* is the MIB object for the current number of TCP connections to this service).
- *cntExt.mib* - Contains content rule objects (for example, *apCntHits* is the MIB object for the total number of hits on this service for this content rule).

Refer to [Chapter 5, Configuring Simple Network Management Protocol \(SNMP\)](#), for information on CSS Enterprise MIBs.

Use the **lookup** command to look up a MIB object and view its description. For example, to view the description for the *apSvcConnections* object, enter:

```
(config-rmonalarm[1])# lookup apSvcConnections
ASN Name:          apSvcConnections
MIB:               svcext
Object Identifier: 1.3.6.1.4.1.2467.1.15.2.1.20
Argument Type:    Integer
Range:            0-4294967295
Description:
    The current number of TCP connections to this service
```

Use the **sample-variable** *mib_object* command to specify the sample variable for this RMON alarm. For example, to define the *apSvcConnections* MIB object for the current number of service connections, enter:

```
(config-rmonalarm[1])# sample-variable apSvcConnections
```

To see a list of SNMP variables, use the **sample-variable ?** command. For example:

```
(config-rmonalarm[1])# sample-variable ?  
  
  apSvcLoadInfoTimeout  
  apSvcLoadSvcStatRptTimeout  
  apSvcLoadEnable  
  apSvcLoadDecayInterval  
  apSvcLoadStepStatic  
  apSvcLoadStepSize  
  apSvcLoadThreshold  
  ...
```

To remove the sample variable, enter:

```
(config-rmonalarm[1])# no sample-variable
```

Defining the Absolute or Delta Sampling Method

When you configure an alarm, you can define the sampling method to compare the sample value of a MIB object to either:

- The configured threshold directly. This sampling is similar to a gauge, recording the value as it fluctuates up and down (see [Figure 6-2](#)).
- The previous sampling, and then their difference is compared to the configured threshold. This sampling is similar to a counter, recording the value that constantly increases (see [Figure 6-3](#)).

Absolute sampling compares the sample value to the configured threshold. For example, if you want to know when 30,000 service connections occur on the CSS during a sample interval, configure the apSvcConnections MIB object with absolute sampling. The apSvcConnections object is the current number of connections on a service. To define an absolute sampling, enter:

```
(config-rmonalarm[1])# sample-type absolute
```

Delta sampling (the default sampling method) compares the current sample value with the previous sample and compares their difference to the configured threshold. For example, if you want to know when the number of content rule hits increase by 100,000 as compared to its previous sampling, configure the apCntHits MIB object with delta sampling. apCntHits is an ever-increasing count of hits.

To define a delta sampling, enter:

```
(config-rmonalarm[1])# sample-type delta
```

To reset the sample type to delta sampling, enter:

```
(config-rmonalarm[1])# no sample-type
```

Defining a Rising Threshold and Rising Event

If you want to be notified when a sampling is greater than or equal to a specific number, set a rising threshold and associate it with a configured RMON event.

You must create an RMON event before you can associate it with an alarm.

For a single rising alarm event to occur, a sampled value is greater than or equal to the rising threshold value, and the value at the last sampling interval is less than this threshold.

- Use the **rising-threshold** *rising_value* command to set the threshold for the alarm. The *rising_value* variable is the threshold for the rising sample type. Enter an integer from 0 (default) to 4294967295.

To set the rising threshold value of 100, enter:

```
(config-rmonalarm[1])# rising-threshold 100
```

To reset the rising threshold to 0, enter:

```
(config-rmonalarm[1])# no rising-threshold
```

- Use the **rising-event** *rising_index* command to associate a configured event to the RMON alarm when the sampled value exceeds the rising threshold value. The *rising_index* variable is the event index used when a rising threshold is crossed. Enter a previously created RMON event index (see the [“Creating an Index for an RMON Event”](#) section). If you enter 0, no event is generated.

To associate the threshold to RMON event 1, enter:

```
(config-rmonalarm[1])# rising-event 1
```

To see a list of RMON events, enter:

```
(config-rmonalarm[1])# rising-event ?
```

To reset the rising event to 0 (no event is generated), enter:

```
(config-rmonalarm[1])# no rising-event
```

Defining a Falling Threshold and Index

If you want to be notified when a sampling is less than or equal to a specific number, set a falling threshold and associate it to a configured event.



Note

You must create an RMON event before you can associate it with an alarm.

For a single falling alarm event to occur, a sampled value is less than or equal to the falling threshold value, and the value at the last sampling interval is greater than this threshold.

- Use the **falling-threshold** *falling_value* command to set the threshold for the alarm. The *falling_value* variable is the threshold for the falling sample type. Enter an integer from 0 (default) to 4294967295.

To set the falling threshold value of 90, enter:

```
(config-rmonalarm[1])# falling-threshold 90
```

To reset the falling threshold to 0, enter:

```
(config-rmonalarm[1])# no falling-threshold
```

- Use the **falling-event** *falling_index* command to associate a configured event to the RMON alarm when the sampled value exceeds the falling threshold value. The *falling_index* variable is the event index used when a falling threshold is crossed. Enter a previously created RMON event index (see the [“Creating an Index for an RMON Event”](#) section). If you enter 0, no event is generated.

To associate the threshold to RMON event 2, enter:

```
(config-rmonalarm[1])# falling-event 2
```

To see a list of RMON events, enter:

```
(config-rmonalarm[1])# falling-event ?
```

To reset the falling event to 0, enter:

```
(config-rmonalarm[1])# no falling-event
```

Defining a Startup Alarm

A startup alarm allows the CSS to generate an alarm when the first sample triggers a falling threshold or rising threshold (default).

- A startup falling alarm occurs when the first sample is less than or equal to the falling threshold. To enable this alarm, enter:

```
(config-rmonalarm[1])# startup-type falling
```

- A startup rising alarm occurs when the first sample is greater than or equal to the rising threshold. To enable this alarm, enter:

```
(config-rmonalarm[1])# startup-type rising
```

- To enable an alarm when either a falling or rising threshold is triggered, enter:

```
(config-rmonalarm[1])# startup-type rising-and-falling
```

To reset the startup alarm to a rising threshold alarm, enter:

```
(config-rmonalarm[1])# no startup-type
```

Defining the Sampling Interval

The sampling interval is the time interval, in seconds, over which the data is sampled and compared with the rising and falling thresholds. Use the **sample-interval** *interval* command to specify the sampling interval for the RMON alarm. The *interval* variable is the number of seconds from 1 to 65535. The default is 300 seconds.

To enter a sampling interval of 60 seconds, enter:

```
(config-rmonalarm[1])# sample-interval 60
```

With delta sampling, set the sampling interval short enough so the sampled variable, which has a tendency to go up and down very fast, does not wrap during a single sampling period.

To reset the sample interval to 300, enter:

```
(config-rmonalarm[1])# no sample-interval
```

Activating an RMON Alarm

After you configure the alarm attributes, you can activate the alarm. However, before you can activate an alarm, you must specify all attributes for the alarm.

To activate the alarm, enter:

```
(config-rmonalarm[1])# active
```

Note the following when activating an RMON alarm, once an RMON alarm is activated and you want to make modifications to certain alarm attributes, you must first suspend the RMON alarm.

Suspending an RMON Alarm

Suspending an RMON alarm deactivates it, allowing you to make changes to its configuration settings. To suspend an alarm, use the **suspend** command.

For example:

```
(config-rmonalarm[1])# suspend
```

Configuring an RMON History

You can configure the operation of the RMON history that periodically samples any CSS Ethernet port for statistical data. All ports are preconfigured with histories for 30-second and 30-minute intervals, and 50 buckets with one sample for each bucket. However, you can create additional histories for a specific port. The creation of an RMON history for a port allows you to configure the time interval to take the sample and the number of samples you want to save.

You can view the statistical information for the history by using the **show rmon-history** command. For more information about viewing the history, see the [“Viewing History”](#) section.

This section includes the following topics:

- [RMON History Configuration Quick Start](#)
- [Creating an Index for an RMON History](#)
- [Deleting an RMON History Index](#)
- [Setting the RMON History Attributes](#)
- [Activating an RMON History Entry](#)

RMON History Configuration Quick Start

[Table 6-3](#) provides a quick overview of the steps required to configure the attributes for an RMON history. Each step includes the CLI command required to complete the task. For a complete description of each feature and all the options associated with the CLI command, see the sections following [Table 6-3](#).

Table 6-3 RMON History Configuration Quick Start

Steps and Possible Settings

1. Create an RMON history from global configuration mode. Enter an integer from 1 to 65535.

```
(config)# rmon-history 5
Create History <5>, [y/n]:y
```

2. Assign the owner who defined and is using the history resources. Enter a maximum of 32 characters.

```
(config-rmonhistory[5])# owner "Boston_Tech_Lab"
```

3. Define the data source object for the Ethernet port. The port is identified by an ifIndex data object identifier.

```
(config-rmonhistory[5])# data-source ifIndex.3
```

4. Define the time interval for the history. Enter a number from 1 to 3600 seconds. The default is 1800 seconds.

```
(config-rmonhistory[5])# interval 60
```

Table 6-3 RMON History Configuration Quick Start (continued)**Steps and Possible Settings**

5. Define the bucket count for the interval. Enter an integer from 1 to 65535. The default is 50.

```
(config-rmonhistory[5])# requested-buckets 25
```

6. Activate the history.

```
(config-rmonhistory[5])# active
```

The following running-configuration example shows the results of entering the commands in [Table 6-3](#).

```
!***** RMON HISTORY *****
rmon-history 5
  owner "Boston_Tech_Lab"
  data-source ifIndex.3
  interval 60
  requested-buckets 25
  active
```

Creating an Index for an RMON History

The RMON history index identifies the history to the CSS. The RMON history index allows you to assign specific configuration attributes to the RMON history index. When you create an RMON history index, you access the configuration mode for that history automatically.

To create a history index, use the **rmon-history** *index* command from any configuration mode except boot configuration mode. The *index* is an integer from 1 to 65534.



Note

The RMON history index 65535 is administratively predefined and cannot be modified. If you enter this index number, a message similar to the following appears: %% Index internally used. Administrative control not allowed.

To create an RMON history identifier 5, access global configuration mode and enter:

```
(config)# rmon-history 5
```

To see a list of existing RMON history configuration identifiers, enter **rmon-history ?**.

After you create the identifier for the history, the prompt changes to (config-rmonhistory[1]). Now you can define the history as described in the [“Setting the RMON History Attributes”](#) section.

Modifying the Attributes for an Existing RMON History Index

If create an RMON history index but have not activated it, you can modify the attributes of the history index (as described in the [“Setting the RMON History Attributes”](#) section). Once the RMON history index is activated, you cannot modify its attributes. You must delete the history index (see the [“Deleting an RMON History Index”](#) section), recreate it, and respecify the alarm index attributes.

Deleting an RMON History Index

If you have an active RMON history index that requires changes to its attributes or you no longer need it, delete the RMON history index. Before you delete a history index that requires changes, note the settings for its attributes.

To delete an RMON history configuration identifier, use the **no rmon-history** command. This command is available in the RMON alarm, RMON event, RMON history, and global configuration modes.

For example, to delete RMON history 5 and its configuration, enter:

```
(config)# no rmon-history 5
Delete History <5>, [y/n]:y
```

After you delete the history identifier to change its attributes, recreate it as described in the [“Creating an Index for an RMON History”](#) section.

Setting the RMON History Attributes

After you create an RMON history or access RMON history configuration mode for an existing inactive alarm, set the RMON history attributes. This section includes the following topics:

- [Defining the Data Object](#)
- [Assigning an Owner](#)
- [Defining the Bucket Count](#)
- [Defining the Bucket Interval](#)

After you set the attributes, activate the history as described in the “[Activating an RMON History Entry](#)” section.

Defining the Data Object

After you create a history, you must associate it with a CSS Fast Ethernet or Gigabit Ethernet port. To define the data object, use the **data-source** *port* command. The *port* is identified by an ifIndex data object identifier. For example, if your CSS has 12 Ethernet ports, they have data object IDs of ifIndex.1 through ifIndex.12. The Ethernet management port has an ID of ifIndex.14.

To define Ethernet port 4, enter:

```
(config-rmonhistory[5])# data-source ifIndex.4
```

To see a list of data object IDs for all of the CSS Ethernet ports, enter:

```
(config-rmonhistory[5])# show interface
```

Assigning an Owner

Define the owner who configured the RMON history and is using the resources assigned to it. Use the **owner** *owner_name* command to define the owner. The *owner_name* variable is a quoted text string with a maximum of 32 characters.

For example, to define an owner named Boston Tech Lab, enter:

```
(config-rmonhistory[5])# owner "Boston Tech Lab"
```

Defining the Bucket Count

You can define a bucket count, which is the number of discrete sampling intervals over which data is saved for a history entry. Use the **requested-buckets** *count* command to define a bucket count. The *count* variable is an integer from 1 to 65535. The default is 50.

To define a bucket count of 25, enter:

```
(config-rmonhistory[5])# requested-buckets 25
```

Defining the Bucket Interval

You can specify the time interval, in seconds, to take a bucket sample for an RMON history operation. Use the **interval** *value* command to set this interval. Enter an integer from 1 to 3600 seconds. The default is 1800 (30 minutes).

To define a time interval of 60 seconds, enter:

```
(config-rmonhistory[5])# interval 60
```

Activating an RMON History Entry

After you configure the history attributes, you can activate the history for the port. Use the **active** command to activate an RMON history entry. Before you activate the history, make sure you finish configuring it and are satisfied with the RMON history settings. After you activate a history, you cannot modify its configuration settings. The only way to change the history is to delete it, and then recreate it.

Before activating this command, you must specify the owner of the RMON history entry.

To activate the history, enter:

```
(config-rmonhistory[5])# active
```

Viewing RMON Information

RMON information includes:

- Ethernet port statistics and history data that you can view from the CSS through **show** commands.
- Alarm event notifications that are sent to log locations on the CSS or an SNMP network management station. Refer to [Chapter 5, Configuring Simple Network Management Protocol \(SNMP\)](#), for information on configuring SNMP on the CSS.

The following sections provide information on:

- [Viewing Statistics](#)
- [Viewing History](#)
- [Viewing Events in a Log File](#)

Viewing Statistics

RMON statistics provide a summary of data received over the Fast Ethernet or Gigabit Ethernet ports. You can view RMON statistics either in a CSS CLI session through the **show rmon** command, the **show ether-errors** command (refer to the *Cisco Content Services Switch Routing and Bridging Configuration Guide*), or directly through an SNMP network management station by using ether-stats MIB objects (refer to RFC 1398).

The CSS **show rmon** command allows you to display the extended 64-bit RMON statistics for a specific Ethernet port or all Ethernet ports in the CSS. The CSS Enterprise ap64Stats MIB defines these statistics. You can also display the RFC 1757 32-bit statistics by adding the **-32** suffix to the **show rmon** command.

To display the RMON statistics for all ports in the CSS, enter:

```
# show rmon
```

To display the RFC 1757 32-bit statistics, enter:

```
# show rmon-32
```

To display the RMON statistics for a specified port in the CSS, enter:

```
# show rmon port_name
```

The *port_name* variable is the name of the physical port (for example, ethernet-4). Enter the *port_name* variable as a case-sensitive, unquoted text string.

To display the RFC 1757 32-bit statistics, enter **show rmon-32** *port_name*.

To see a list of ports, enter:

```
# show rmon ?
```

To display the extended RMON statistics for the Ethernet-4 port in the CSS, enter:

```
# show rmon ethernet-4
```

[Table 6-4](#) lists and describes the fields in the **show rmon** command output.

Table 6-4 *Field Descriptions for the show rmon Command*

Field	Description
Bytes	The total number of received bytes.
Packets	The total number of received packets (including bad packets, broadcast packets, and multicast packets).
Broadcast Packets	The total number of good received packets that were directed to the broadcast address. The number of broadcast packets does not include multicast packets.
Multicast Packets	The total number of good received packets that were directed to a multicast address. The number of multicast packets does not include packets directed to the broadcast address.
CRC Alignment Errors	The total number of packets received that had a length (excluding framing bits, but including FCS octets) from 64 and 1518 octets, but had an FCS Error, a bad Frame Check Sequence (FCS) with an integral number of octets, or an Alignment Error, a bad FCS with a non-integral number of octets.
Oversize Packets	The total number of received packets that were longer than 1518 octets (excluding framing bits, but including FCS octets) and were otherwise well formed.

Table 6-4 *Field Descriptions for the show rmon Command (continued)*

Field	Description
Undersize Packets	The total number of received packets that were less than 64 octets long (excluding framing bits, but including FCS octets) and were otherwise well-formed.
Fragments	<p>The total number of received packets that were less than 64 octets in length (excluding framing bits but including FCS octets) and had an FCS Error, a bad Frame Check Sequence (FCS) with an integral number of octets, or an Alignment Error, a bad FCS with a non-integral number of octets.</p> <p>It is normal for fragment statistics to increment because the CSS counts both runts (which are normal occurrences due to collisions) and noise hits.</p>
Drop Events	The total number of events in which packets were dropped by the probe due to lack of resources. This number is not necessarily the number of packets dropped; it is the number of times this condition has been detected.
Slobbers	An internal counter. This field is always zero.
Jabbers	<p>The total number of packets received that were longer than 1518 octets (excluding framing bits, but including FCS octets), and had an FCS Error, a bad Frame Check Sequence (FCS) with an integral number of octets, or Alignment Error, a bad FCS with a non-integral number of octets.</p> <p>This definition of jabber is different than the definition in IEEE-802.3, section 8.2.1.5, 10BASE5, and section 10.3.1.4, 10BASE2. These documents define jabber as the condition where any packet exceeds 20 ms. The allowed range to detect jabber is between 20 and 150 ms.</p>

Table 6-4 *Field Descriptions for the show rmon Command (continued)*

Field	Description
Collisions	<p>The best estimate of the total number of collisions on this Ethernet segment.</p> <p>The returned value depends on the location of the RMON probe. Section 8.2.1.3, 10BASE-5, and section 10.3.1.3, 10BASE-2, of IEEE standard 802.3 states that a station must detect a collision, in the receive mode, if three or more stations are transmitting simultaneously. A repeater port must detect a collision when two or more stations are transmitting simultaneously. Thus, a probe placed on a repeater port might record more collisions than would a probe connected to a station on the same segment.</p> <p>Probe location plays a much smaller role when considering 10BASE-T. IEEE standard 802.3 14.2.1.4, 10BASE-T, defines a collision as the simultaneous presence of signals on the DO and RD circuits (transmitting and receiving at the same time). A 10BASE-T station can detect collisions only when it is transmitting. Probes placed on a station and a repeater should report the same number of collisions.</p> <p>Ideally, an RMON probe inside a repeater should report collisions between the repeater and one or more other hosts (transmit collisions as defined by IEEE 802.3k), plus receiver collisions observed on any coaxial segments to which the repeater is connected.</p>
Packets (0-64) Packets (65-127) Packets (128-255) Packets (256-511) Packets (512-1023) Packets (1024-1518)	<p>The total number of received packets (including bad packets) that were between the following octets in length inclusive (excluding framing bits but including FCS octets):</p> <ul style="list-style-type: none"> • 0 to 64 • 65 to 127 • 128 to 255 • 256 to 511 • 512 to 1023 • 1024 to 1518

Clearing RMON Statistics

Use the **clear statistics** *port_name* command to reset the RMON statistics on a CSS Ethernet port to zero. The *port_name* variable is the name of the physical port (for example, ethernet-4). Enter the *port_name* variable as a case-sensitive, unquoted text string.

To clear the statistics for Ethernet port 1, enter:

```
# clear statistics Ethernet-1
```

To see a list of ports, enter:

```
# clear statistics ?
```

**Note**

When you reset RMON statistics on a CSS Ethernet port to zero, the Ethernet errors and MIB-II statistics for the port are also reset to zero.

Viewing History

You can display the default and configured RMON history information for a specific Ethernet port or all Ethernet ports in the CSS. For information on configuring an RMON history, see the [“Configuring an RMON History”](#) section.

By default, the CSS maintains two tables of history statistics for each port. One table contains the last 50 samples at 30-second intervals. The other table contains 50 samples at 30-minute intervals. You cannot modify the configuration for these histories.

- To view the RMON history for all ports in the CSS, enter:

```
# show rmon-history
```

- To display the RMON history for a specified port, enter:

```
# show rmon-history port_name
```

To see a list of ports in the CSS, enter:

```
# show rmon-history ?
```

- To display the RMON history for a specified port and history index, enter:

```
# show rmon-history port_name history_index
```

To view the history 5 for the Ethernet-4 port, enter:

```
# show rmon-history ethernet-4 5
```

To see a list of history indexes associated with a specified port, enter:

```
# show rmon-history port_name ?
```

To see a list of histories for the Ethernet-4 port, enter:

```
# show rmon-history ethernet-4 ?
```

Table 6-5 lists and describes the fields in the **show rmon-history** command output.

Table 6-5 *Field Descriptions for the show rmon-history Command*

Field	Description
Owner	The owner who configured the entry and is using the resources assigned to it.
Start Time	The time when the bucket sampling started.
Interval	The time interval, in seconds, when RMON takes a bucket sample.
Buckets	The number of discrete sampling intervals over which data is saved for the history.
Time	The time that the sample was taken.
Sample	The number of the sample.

Table 6-5 Field Descriptions for the show rmon-history Command (continued)

Field	Description
Octets	<p>The total number of octets of data (including those in bad packets) received on the network, excluding framing bits but including FCS octets.</p> <p>You can use this object as a reasonable estimate of Ethernet utilization. If greater precision is desired, sample the Ethernet statistic packet and octet objects before and after a common interval. The differences in the sampled values are packets (Pkts) and Octets, respectively, and the number of seconds in the Interval. These values are used to calculate the utilization of a 10 Mbps Ethernet port as follows:</p> $\text{Utilization} = \frac{\text{Pkts} * (9.6 + 6.4) + (\text{Octets} * .8)}{\text{Interval} * 10,000}$ <p>The result of this equation is the utilization value, which is the utilization percentage of the Ethernet segment on a scale of 0 to 100 percent.</p>
Packets	The total number of received packets (including bad packets, broadcast packets, and multicast packets).
Errors	The total number of errors that RMON received for this port.
Util%	The bandwidth utilization percentage of the Ethernet segment on a scale of 0 to 100 percent.

Viewing Events in a Log File

The CSS can send notifications of RMON alarm events to a traplog file or a configured log location, such as a log file on the CSS disk, a CSS session, a host syslog daemon, or an e-mail address. The notification itself displays the time that the event occurred, the event number, and its configured description in parentheses.

For example:

```
FEB 15 15:41:22 EVENT#4 FIRED: (Service Toys exceeded 30,000
connections).
```

For information on configuring an RMON event, see the “[Configuring an RMON Event](#)” section. For information on configuring an RMON alarm, see the “[Configuring an RMON Alarm](#)” section.

Viewing a Traplog File

A traplog file is an ASCII file in the log directory containing generic and enterprise SNMP traps. No configuration is necessary for the traplog file. When an RMON alarm event occurs, a notification of its occurrence is automatically saved in the trap log file on the CSS. Even when traps are disabled, the CSS still produces a log message for any event that would normally generate a trap.

When a traplog file reaches its maximum size (50 MB for a hard disk-based CSS, 10 MB for a flash disk-based CSS), the CSS renames the traplog file to traplog.prev as a backup file and starts a new traplog file. The CSS overwrites the backup traplog file when it renames the traplog file. Each time the CSS reboots, it continues to use the existing traplog file until it reaches its maximum size.

The traps sent to the traplog file are the same traps sent to an SNMP network management station. Refer to [Chapter 5, Configuring Simple Network Management Protocol \(SNMP\)](#), for information on configuring SNMP.

To display all SNMP traps that have occurred on the CSS, enter:

```
# show log traplog
```

Viewing a CSS Disk Log File

Before the CSS can send an event to a log location, you must:

- Configure the location by using the **logging disk, host, line, or sendmail** command.
- Enable logging for the network management subsystem. To do so, enter:

```
(config)# logging subsystem netman level info-6
```

Refer to [Chapter 4, Using the CSS Logging Features](#), for details on configuring logging for the CSS.

To view the events in a log file on the CSS disk, use the **show log log_filename** command. To view a log file named log1, enter:

```
# show log log1
```