

EEM Event Registration Tcl Command Extensions

The following conventions are used for the syntax documented on the Tcl command extension pages:

- An optional argument is shown within square brackets, for example:

[type ?]

- A question mark ? represents a variable to be entered.
- Choices between arguments are represented by pipes, for example:

priority low|normal|high

Note For all EEM Tcl command extensions, if there is an error, the returned Tcl result string contains the error information.

Note Arguments for which no numeric range is specified take an integer from -2147483648 to 2147483647, inclusive.

- event_register_appl, on page 2
- event_register_cli, on page 4
- event_register_counter, on page 7
- event_register_gold, on page 9
- event_register_identity, on page 15
- event_register_interface, on page 17
- event_register_ioswdsysmon, on page 22
- event_register_ipsla, on page 25
- event_register_mat, on page 28
- event_register_neighbor_discovery, on page 30
- event_register_nf, on page 33
- event_register_none, on page 36
- event_register_oir, on page 38
- event_register_process, on page 40
- event_register_resource, on page 43
- event_register_rf, on page 44

- event_register_routing, on page 47
- event_register_rpc, on page 49
- event_register_snmp, on page 51
- event_register_snmp_notification, on page 55
- event_register_snmp_object, on page 57
- event_register_syslog, on page 60
- event_register_timer, on page 62
- event_register_timer_subscriber, on page 66
- event_register_track, on page 68
- event_register_wdsysmon, on page 70

event_register_appl

Registers for an application event. Use this Tcl command extension to run a policy when an application event is triggered following another policy's execution of an **event_publish** Tcl command extension; the **event_publish** command extension publishes an application event.

In order to register for an application event, a subsystem must be specified. Either a Tcl policy or the internal Embedded Event Manager (EEM) API can publish an application event. If the event is being published by a policy, the sub_system argument that is reserved for a policy is 798.

Syntax

```
event_register_appl [tag ?] sub_system ? type ? [queue_priority low|normal|high|last] [maxrun
 ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
sub_system	(Mandatory) Number assigned to the EEM policy that published the application event. The number is set to 798 because all other numbers are reserved for Cisco use. If this argument is not specified, all components are matched.
type	(Mandatory) Event subtype within the specified event. The sub_system and type arguments uniquely identify an application event. If this argument is not specified, all types are matched. If you specify this argument, you must choose an integer between 1 and 4294967295, inclusive. There must be a match of component and type between the **event_publish** command extension and the **event_register_appl**command extension in order for the publishing and registration to work.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If multiple conditions exist, the application event will be raised when all the conditions are satisfied.

Result String

None

Set _cerrno

No

Event_reqinfo

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"sub_system 0x%x type %u data1 {%s} data2 {%s} data3 {%s} data4 {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.

Event Type	Description
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec **event_pub_msec**	The time, in seconds and milliseconds, when the event was published to the Embedded Event Manager (EEM).
sub_system	Number assigned to the EEM policy that published the application event. Number is set to 798 because all other numbers are reserved for Cisco use.
type	Event subtype within the specified component.
data1 data2 data3 data4	Argument data that is passed to the application-specific event when the event is published. The data is character text, an environment variable, or a combination of the two.

event_register_cli

Registers for a CLI event. Use this Tcl command extension to run a policy when a CLI command of a specific pattern is entered based on pattern matching performed against an expanded CLI command.



Note The user can enter an abbreviated CLI command, such as **sh mem summary**, and the parser will expand the command to **show memory summary**to perform the matching.



Note The functionality provided in the CLI event detector only allows a regular expression pattern match on a valid IOS CLI command itself. This does not include text after a pipe character when redirection is used.

Syntax

```
event_register_cli [tag ?] sync yes|no skip yes|no
[occurs ?] [period ?] pattern ? [default ?] [enter] [questionmark] [tab] [mode]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
sync	(Mandatory) A "yes" means that the policy (the event publish) will run synchronously with the CLI command; a "no" means that the event publish will be performed asynchronously with the CLI command. The event detector will be notified when the policy completes running. The exit status of the policy indicates whether or not the CLI command should be executed: if the exit status is zero, which means that the policy is executed successfully, the CLI command will not be executed; otherwise, the CLI command will be executed.

skip	Mandatory if the sync argument is "no" and should not exist if the sync argument is "yes." If the skip argument is "yes," it means that the CLI command should not be executed. If the skip argument is "no," it means that the CLI command should be executed. **Caution** When the skip argument is "yes," unintended results may be produced if the pattern match is made for configuration commands because the CLI command that matches the regular expression will not be executed.
occurs	(Optional) The number of occurrences before the event is raised. If this argument is not specified, the event is raised on the first occurrence. If this argument is specified, it must be an integer between 1 and 4294967295, inclusive.
period	(Optional) Specifies a backward looking time window in which all CLI events must occur (the occurs clause must be satisfied) in order for an event to be published (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent event is used.
pattern	(Mandatory) Specifies the regular expression used to perform the CLI command pattern match.
default	(Optional) The time period during which the CLI event detector waits for the policy to exit (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to run the command. If this argument is not specified, the default time period is set to 30 seconds.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
enter	(Optional) Specifies to perform the event match when the user presses the Enter key. When this parameter is used, the input string will not be expanded before matching.

questionmark	(Optional) Specifies to perform the event match when the user presses the ? key. When this parameter is used, the input string will not be expanded before matching.
tab	(Optional) Specifies to perform the event match when the user presses the Tab key. When this parameter is used, the input string will not be expanded before matching.
mode	(Optional) Events will only be generated when the parser is in the specified parser mode. The available modes can be listed using the **show parser dump** CLI command. The mode parameter is checked when any one of the optional parameters--enter, questionmark, or tab--is specified.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If multiple conditions are specified, the CLI event will be raised when all the conditions are matched.

Result String

None

Set _cerrno

No



Note This policy runs before the CLI command is executed. For example, suppose policy_CLI is registered to run when the **copy** command is entered. When the **copy** command is entered, the CLI event detector finds a pattern match and triggers this policy to run. When the policy execution ends, the CLI event detector determines if the **copy** command needs to be executed according to "sync", "skip" (set in the policy), and the exit status of the policy execution if needed.

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u msg {%s} msg_count %d line %u key %u tty %u error_code %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec **event_pub_msec**	The time, in seconds and milliseconds, at which the event was published to the EEM.
event_severity	The severity of the event.
msg	Text entered at the CLI prompt.
msg_count	Number of times the pattern matched before the event was triggered.
line	The text the parser was able to expand up to the point where the matched key was entered.
key	The enter, questionmark, or tab key.
tty	Corresponds to the line number the user is executing the command on.
error_code	The error code in CLI. 0 --No error from parser up to point where a key was entered. 1--Command is ambiguous up to point where a key was entered. 4--Unknown command up to point where a key was entered.

event_register_counter

Registers for a counter event as both a publisher and a subscriber. Use this Tcl command extension to run a policy on the basis of a named counter crossing a threshold. This event counter, as a subscriber, identifies the name of the counter to which it wants to subscribe and depends on another policy or another process to actually manipulate the counter. For example, let policyB act as a counter policy, whereas policyA (although it does not need to be a counter policy) uses **register_counter**, **counter_modify**, or **unregister_counter** Tcl command extensions to manipulate the counter defined in policyB.

Syntax

```
event_register_counter [tag ?] name ? entry_op gt|ge|eq|ne|lt|le entry_val ?
exit_op gt|ge|eq|ne|lt|le exit_val ? [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
name	(Mandatory) Name of the counter.
entry_op	(Mandatory) Entry comparison operator used to compare the current counter value with the entry value; if true, an event will be raised and event monitoring will be disabled until exit criteria are met.

entry_val	(Mandatory) Value with which the current counter value should be compared to decide if the counter event should be raised.
exit_op	(Mandatory) Exit comparison operator used to compare the current counter value with the exit value; if true, event monitoring for this event will be reenabled.
exit_val	(Mandatory) Value with which the current counter value should be compared to decide if the exit criteria are met.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"name {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
name	Counter name.

event_register_gold

Registers for a Generic Online Diagnostic (GOLD) failure event. Use this Tcl command extension to run a policy on the basis of a Generic Online Diagnostic (GOLD) failure event for the specified card and subcard.

Syntax

```
event_register_gold card all|card_number
[subcard all|subcard_number]
[new_failure TRUE|FALSE]
[severity_major TRUE]
[severity_minor TRUE]
[severity_normal TRUE]
[action_notify TRUE|FALSE]
[testing_type [bootup|ondemand|schedule|monitoring]]
[test_name [testname]]
[test_id [testnumber]]
[consecutive_failure consecutive_failure_number]
[platform_action [action_flag]]
[maxrun ?]
[queue_priority low|normal|high|last]
[nice 0|1]
```

Arguments

card	(Mandatory) Specifies whether all cards or one card is to be monitored: • card all--Specifies that all cards are to be monitored. This is the default. • card-number--Specifies that the card identified by the number card-number is to be monitored. This argument must be specified to complete the **event_register_gold**Tcl command extension.

subcard	(Optional) Specifies that one or more subcards are to be monitored: • subcard all--Specifies that all subcards are to be monitored. • subcard-number--Specifies that the subcard identified by the number subcard-number is to be monitored. If this argument is not specified, all subcards are monitored by default.
new_failure	(Optional) Specifies event criteria based on the new test failure information from GOLD: • new_failure TRUE--Specifies that the event criterion for the new test failure is true from GOLD. • new_failure FALSE--Specifies that the event criterion for the new test failure is false from GOLD. If this argument is not specified, the new test failure information from GOLD is not considered in the event criteria.
severity_major	(Optional) Specifies that the event criteria for diagnostic result matches with the diagnostic major error from GOLD.
severity_minor	(Optional) Specifies that the event criteria for diagnostic result matches with diagnostic minor error from GOLD.
severity_normal	(Optional) Specifies that the event criteria for diagnostic result matches with diagnostic normal from GOLD. This is the default.
action_notify	(Optional) Specifies the event criteria based on the action notify information from GOLD: • action_notify TRUE--Specifies that the event criterion for the action notify is true from GOLD. • action_notify FALSE--Specifies that the event criterion for the action notify is false from GOLD. If this argument is not specified, the action notify information from GOLD is not considered in the event criteria.
testing_type	(Optional) Specifies the event criteria based on the testing types of the diagnostic from GOLD: • testing_type bootup--Specifies the diagnostic tests that are running on system bootup. • testing_type ondemand--Specifies the diagnostic tests that are running from CLI after the card is online. • testing_type schedule--Specifies the scheduled diagnostic tests. • testing_type monitoring--Specifies the diagnostic tests that are running periodically in the background to monitor the health of the system. If this argument is not specified, the testing type information from GOLD is not considered in the event criteria and the policy applies to all the diagnostic testing types.

test_name	(Optional) Specifies the event criteria based on the test name: • test_name test-name--Specifies the event criteria based on the test with the name test-name. If this argument is not specified, the test name information from GOLD is not considered in the event criteria.
test_id	(Optional) Specifies the event criteria based on test ID: • test_id test-id--Specifies the event criteria based on the test with the ID number test-id. The maximum value of test-id is 65535. **Note** Because the test ID can be different for the same test on different line cards, usually the test_name keyword should be used instead. If the test ID is specified and conflicts with the specified test name, the test name overwrites the test ID. If this argument is not specified, test ID information from GOLD is not considered in the event criteria.
consecutive_failure	(Optional) Specifies the event criteria based on consecutive test failure information from GOLD: • consecutive_failure consecutive-failure-number--Specifies that the event criterion is based on the occurrence of consecutive-failure-number consecutive test failures. If this argument is not specified, consecutive test failure information from GOLD is not considered in the event criteria.
platform_action	(Optional) Specifies whether callback to the platform is needed when all the event criteria are matched. When callback is needed, the platform needs to register a callback function through the provided registry. • platform_action action-flag-number--Specifies that, when callback to the platform is needed, specific information is specified by the platform-specific action-flag-number value. The maximum value of action-flag-number is 65535. **Note** It is up to the platform to determine what action needs to be taken based on the flag. If this argument is not specified, there is no callback.
maxrun	(Optional) Specifies the maximum runt time of the script. • maxrun max-run-time-number--Specifies that the maximum run time of the script is max-run-time-number seconds. The maximum value of max-run-time-number is 4294967295 seconds. If this argument is not specified, the default run time is 20 seconds.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
nice	(Optional) Policy run-time priority setting: • nice 0--Specifies that the policy is run at the default run-time priority level. • nice 1--Specifies that the policy is run at a run-time priority that is less than the default priority. If this argument is not specified, the default run-time priority is used.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u card %u sub_card %u"
"event_severity {%s} event_pub_sec %u event_pub_msec %u overall_result %u"
"new_failure {%s} action_notify {%s} tt %u tc %u bl %u ci %u pc %u cn {%s}"
"sn {%s} tn# {%s} ta# %s ec# {%s} rc# %u lf# {%s} tf# %u cf# %u tr# {%s}"
"tr#p# {%s} tr#d# {%s}"
```

Event Type	Description
action_notify	Action notify information in GOLD event: true or false.

Event Type	Description
bl	The boot-up diagnostic level, which can be one of the following values: • 0: complete diagnostic • 1: minimal diagnostics • 2: bypass diagnostic
card	Card information for the GOLD event.
cf *testnum*	Consecutive failure, where *testnum* is the test number. For example, **cf3** is the EEM built-in environment variable for consecutive failure of test 3.
ci	Card index.
cn	Card name.
ec *testnum*	Test error code, where *testnum* is the test number. For example, **ec3** is the EEM built-in environment variable for the error code of test 3.
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_pub_msec **event_pub_sec**	The time, in milliseconds and seconds, when the event was published to the EEM.
event_severity	GOLD event severity, which can be one of the following values: • normal • minor • major.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
lf *testnum*	Last fail time, where *testnum* is the test number. For example, **lf3** is the EEM built-in variable for the last fail time of test 3. The timestamp format is *mmm dd yyyy hh*:*mm*:*ss*. For example, Mar 11 1960 08:47:00.
new_failure	The new test failure information in a GOLD event flag: true or false.
overall_result	The overall diagnostic result, which can be one of the following values: • 0: OK • 3: minor error • 4: major error • 14: unknown result

Event Type	Description
pc	Port counts.
rc *testnum*	Test total run count, where *testnum* is the test number. For example, **rc3** is the EEM built-in variable for the total run count of test 3.
sn	Card serial number.
sub_card	The subcard on which a GOLD failure event was detected.
ta *testnum*	Test attribute, where *testnum* is the test number. For example, **ta3** is the EEM built-in variable for the test attribute of test 3.
tc	Test counts.
tf *testnum*	Total failure count, where *testnum* is the test number. For example, **tf3** is the EEM built-in variable for the total failure count of test 3.
tn *testnum*	Test name, where *testnum* is the test number. For example, **tn3** is the EEM built-in variable for the name of test 3.
tr *testnum*	Test result, where *testnum* is the test number. For example, **tr6** is the EEM built-in variable for test 6 where test 6 is not a per-port test and not a per-device test. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
tr *testnum* **d** *devnum*	Per-device test result, where *testnum* is the test number and *devnum* is the device number. For example, **tr3d20** is the EEM built-in variable for the test result for test 3, device 20. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown
tr *testnum* **p** *portnum*	Per-port test result, where *testnum* is the test number and *portnum* is the device number. For example, **tr5p20** is the EEM built-in variable for the test result for test 3, port 20. The test result is one of the following values: • P: diagnostic result Pass • F: diagnostic result Fail • U: diagnostic result Unknown

Event Type	Description
tt	The testing type, which can be one of the following: • 1: A boot-up diagnostic • 2: An on-demand diagnostic • 3: A schedule diagnostic • 4: A monitoring diagnostic

event_register_identity

Registers for an identity event. Use this Tcl command extension to generate an event when AAA authentication or authorization is successful or failure or after normal user traffic on the port is allowed to flow.

Syntax

```
event_register_identity [tag ?] interface ?
[aaa-attribute ?]
[authc {all | fail | success}]
[authz {all | fail | success}]
[authz-complete]
[mac-address ?]
[queue_priority {normal | low | high | last}]
[maxrun ?] [nice {0 | 1}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
interface	A regular expression pattern to match against interface names.
aaa-attribute	(Optional) A regular expression that can be used to filter events by specific AAA attributes.
authc	(Optional) Triggers events on successful, failed or both successful and failed authentication.
authz	(Optional) Triggers events on successful, failed or both successful and failed authorization.
authz-complete	(Optional) Triggers events once the device connected to the interface is fully authenticated, authorized and normal traffic has begun to flow on that interface.
mac-address	(Optional) A regular expression pattern that can be used to filter events by mac addresses of the remote device.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo For EEM_EVENT_IDENTITY

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u identity_stage %u identity_status %u interface %u identity_mac %u
identity_<attribute> {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.
event_severity	The severity of the event.

Event Type	Description
identity_stage	One among authentication, autherization or autherization-complete stages.
identity_status	Success or one of these failure types: fail_authc, fail_aaa_server, fail_no_response, fail_timeout, fail_authz. For autherization-complete it is always success.
interface	The interface for the event.
identity_mac	The MAC address of the remote device for the event.
identity_<attribute>	For each AAA attribute, a set a dynamic variable to the value corresponding to that AAA attribute in the attribute or value list.

event_register_interface

Registers for an interface counter event. Use this Tcl command extension to generate an event when specified interface counters exceed specified thresholds.

Syntax

```
event_register_interface [tag ?] name ?
parameter ? entry_op gt|ge|eq|ne|lt|le
entry_val ? entry_val_is_increment TRUE|FALSE
entry_type value|increment|rate
[exit_comb or|and]
[exit_op gt|ge|eq|ne|lt|le]
[exit_val ?] [exit_val_is_increment TRUE|FALSE]
[exit_type value|increment|rate]
[exit_time ?] [poll_interval ?]
[average_factor ?] [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
name	(Mandatory) The name of the interface being monitored, for example, Ethernet 0/0. Abbreviations and spaces are not allowed.

parameter	(Mandatory) The name of the counter being compared as follows: • input_errors--Includes runts, giants, no buffer, CRC, frame, overrun, and ignored counts. Other input-related errors can also cause the input errors count to be increased, and some datagrams may have more than one error; therefore, this sum may not balance with the sum of enumerated input error counts. • input_errors_crc--Cyclic redundancy checksum generated by the originating LAN station or far-end device does not match the checksum calculated from the data received. • input_errors_frame--Number of packets received incorrectly having a CRC error and a noninteger number of octets. • input_errors_overrun--Number of times the receiver hardware was unable to hand received data to a hardware buffer because the input rate exceeded the receiver's ability to handle the data. • input_packets_dropped--Number of packets dropped because of a full input queue. • interface_resets--Number of times that an interface has been completely reset. • output_buffer_failures--Number of failed buffers and number of buffers swapped out. • output_buffer_swappedout--Number of packets swapped to DRAM.

parameter (continued)	• output_errors--Sum of all errors that prevented the final transmission of datagrams out of the interface being examined. Note that this may not balance with the sum of the enumerated output errors, because some datagrams may have more than one error, and others may have errors that do not fall into any of the specifically tabulated categories. • output_errors_underrun--Number of times that the transmitter has been running faster than the device can handle. • output_packets_dropped--Number of packets dropped because of a full output queue. • receive_broadcasts--Number of broadcast or multicast packets received by the interface. • receive_giants--Number of packets that are discarded because they exceed the maximum packet size of the medium. • receive_rate_bps--Interface receive rate in bytes per second. • receive_rate_pps--Interface receive rate in packets per second. • receive_runts--Number of packets that are discarded because they are smaller than the minimum packet size of the medium. • receive_throttle--Number of times that the receiver on the port was disabled, possibly because of buffer or processor overload. • reliability--Reliability of the interface as a fraction of 255 (255/255 is 100 percent reliability), calculated as an exponential average over 5 minutes. • rxload--Receive rate of the interface as a fraction of 255 (255/255 is 100 percent). • transmit_rate_bps--Interface transmit rate in bytes per second. • transmit_rate_pps--Interface transmit rate in packets per second. • txload--Transmit rate of the interface as a fraction of 255 (255/255 is 100 percent).
entry_op	(Mandatory) The comparison operator used to compare the current interface value with the entry value; if true, an event will be raised and event monitoring will be disabled until exit criteria are met.
entry_val	(Mandatory) The value at which the event will be triggered.
entry_val_is_increment	(Mandatory) If TRUE, the entry_val field is treated as an incremental difference and is compared with the difference between the current counter value and the value when the event was last true (the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. If FALSE, the entry_val field is compared against the current counter value. **Note** This keyword has been deprecated, and if specified, the syntax is converted into equivalent entry-type keyword syntax.

entry-type	Specifies a type of operation to be applied to the object ID specified by the entry-val argument. Value is defined as the actual value of the entry-val argument. Increment uses the entry-val field as an incremental difference and the entry-valis compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_comb	(Optional) Used to indicate the combination of exit condition tests required to rearm the event trigger; if the and operator is specified, both exit value and exit time tests must be true to cause rearm; if the or operator is specified, either exit value or exit time tests can be true to cause event monitoring to be rearmed.
exit_op	(Optional) The comparison operator used to compare the current interface value with the exit value; if true, event monitoring for this event will be reenabled.
exit_val	(Optional) The value at which the event is rearmed to be monitored again.
exit_val_is_increment	(Optional) If TRUE, the exit_val field is treated as an incremental difference and is compared with the difference between the current counter value and the value when the event was last true. A negative value checks the incremental difference for a counter that is decreasing. If FALSE, the exit_val field is compared against the current counter value. **Note** In Cisco IOS Release 12.4(20)T, this keyword is deprecated, and if specified, the syntax is converted into equivalent exit-type keyword syntax.
exit-type	(Optional) Specifies a type of operation to be applied to the object ID specified by the exit-val argument. If not specified, the value is assumed. Value is defined as the actual value of the exit-val argument. Increment uses the exit-val field as an incremental difference and the exit-val is compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.

exit_time	(Optional) The time period at which the event is rearmed to be monitored again (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).
poll_interval	(Optional) The frequency used to collect the samples (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 60 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). The poll interval value must not be less than 1 second. The default is 1 second.
average-factor	(Optional) Number in the range from 1 to 64 used to calculate the period used for rate-based calculations. The average-factor value is multiplied by the poll-interval value to derive the period in milliseconds. The minimum average factor value is 1.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event_severity {%s} name {%s} parameter {%s} value %d"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec **event_pub_msec**	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	Interface event severity, which can be one of the following values: • normal • minor • major
name	Name of the interface.
parameter	Name of the parameter.
value	The incremental/decremental difference compared to the last event triggered or the absolute value of the parameter being monitored, depending on the specified value of entry_val_is_increment.

event_register_ioswdsysmon

Registers for an IOSWDSysMon event. Use this Tcl command extension to generate an event when a Cisco IOS task exceeds specific CPU utilization or memory thresholds. A Cisco IOS task is called a Cisco IOS process in native Cisco IOS.

Syntax

```
event_register_ioswdsysmon [tag ?] [timewin ?] [sub12op and|or] [sub1 ?] [sub2 ?]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
timewin	(Optional) Defines the time window within which all of the subevents must occur in order for an event to be generated (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).
sub12_op	(Optional) The combination operator for comparison between subevent 1 and subevent 2.
sub1	(Optional) The subevent 1 specification.
sub2	(Optional) The subevent 2 specification.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Subevent Syntax

```
cpu_proc path ? taskname ? op gt|ge|eq|ne|lt|le val ? [period ?]
mem_proc path ? taskname ? op gt|ge|eq|ne|lt|le val ? [is_percent TRUE|FALSE] [period ?]
```

Subevent Arguments

cpu_proc	(Mandatory) Specifies the use of a sample collection of CPU statistics.

path	(Mandatory) Software Modularity images only. The pathname of the POSIX process that contains the Cisco IOS scheduler to be monitored. For example, /sbin/cdp2.iosproc.
taskname	(Mandatory) The name of the Cisco IOS task to be monitored.
op	(Mandatory) The comparison operator used to compare the collected usage sample with the specified value; if true, an event will be raised.
val	(Mandatory) The value to be compared.
period	(Optional) The elapsed time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.
mem_proc	(Mandatory) Specifies the use of a sample collection of memory statistics.
is_percent	(Optional) Whether the specified value is a percentage.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"num_subs %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
num_subs	Number of subevents.

Where the subevent info string is for a CPU_UTIL subevent,

```
"{type %s procname {%s} pid %u taskname {%s} taskid %u value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of subevent.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent.
taskname	Cisco IOS task name for this subevent.
taskid	Cisco IOS task ID for this subevent.
value	Actual average CPU utilization over the measured interval.
sec , **msec**	Elapsed time period for this measured interval.

Where the subevent info string is for a MEM_UTIL subevent,

```
"{type %s procname {%s} pid %u taskname {%s} taskid %u is_percent %s value %u diff %d"
"sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of subevent.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent.
taskname	Cisco IOS task name for this subevent.
taskid	Cisco IOS task ID for this subevent.
is_percent	TRUE or FALSE depending on whether the value is a percentage value.
value	Total memory use in KB or the actual average memory utilization for this measured interval.
diff	The percentage difference between the oldest sample in the measured interval and the latest sample; a negative value represents a decrease.
sec , **msec**	Elapsed time period for this measured interval.

event_register_ipsla

Registers for an event that is triggered by the **event ipsla**command. Use this Tcl command to publish an event when an IPSLA reaction is triggered. The group ID or the operation ID is required to register the event.

Syntax

```
event_register_ipsla [tag ?] group_name ? operation_id ? [reaction_type ?]
[dest_ip_addr ?][queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
group_name	(Mandatory) Specifies the IP SLAs group name.
operation_id	(Mandatory) Specifies the IP SLA operation ID. Number must be in the range from 1 to 2147483647.
reaction_type	(Optional) Specifies the reaction to be taken for the specified IP SLAs operation. Type of IP SLAs reaction--One of the following keywords can be specified: **connectionLoss**, **icpif**, **jitterAvg**, **jitterDSAvg**, **jitterSDAvg**, **maxOfNegativeDS**, **maxOfNegativeSD**, **maxOfPositiveDS**, **maxOfPositiveSD**, **mos**, **packetLateArrival**, **packetLossDS**, **packetLossSD**, **packetMIA**, **packetOutOfSequence**, **rtt**, **timeout**or **verifyError** can be specified. Type of IP SLAs reaction. One of the following keywords can be specified: • connectionLoss • icpif • jitterAvg • jitterDSAvg • jitterSDAvg • maxOfNegativeDS • maxOfNegativeSD • maxOfPositiveDS • maxOfPositiveSD • mos • packetLateArrival • packetLossDS • packetLossSD • packetMIA • packetOutOfSequence • rtt • timeout • verifyError

dest_ip_address	(Optional) Specifies the destination IP address of the destination port for which the IP SLAs events are monitored.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

"event_ID %u event_type %u event_pub_sec %u event_pub_msec %u event_severity %u" "group_name %u operation_id %u condition %u reaction_type %u dest_ip_addr %u" "threshold_rising %u threshold_falling%u measured_threshold_value %u" "threshold_count1 %u threshold count2 %u"

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	The type of event to monitor for the create, update, and delete flow.

Event Type	Description
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
group_name	The name of theIPSLA group.
operation_id	The IPSLA operation ID.
condition	The condition of IPSLA, which can be one of the following: • cleared • occurred
reaction_type	The IPSLA reaction type.
dest_ip_address	The IPSLA destination IP address.
threshold rising	The IPSLA configured rising threshold value.
threshold falling	The IPSLA configured falling threshold value.
measured_threshold_value	The measured threshold value of the IPSLA operation.
threshold_count1	Corresponds to the argument of the threshold type1.
threshold_count2	Corresponds to the argument of the threshold type2.

event_register_mat

Registers for a MAT event. Use this Tcl command extension to generate an event when a mac-address is learned in the mac-address-table.

Syntax

```
event_register_identity [tag ?] interface ?
[mac-address ?]
[type {add | delete}]
[hold-down ?]
[maxrun ?]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
interface	A regular expression pattern to match against interface names.

mac-address	Mandatory if the interface parameter is not specified. A regular expression pattern that can be used to filter events by mac addresses of the remote device.
type	(Optional) Filter based on a mac-address-table event type of add or delete. If not specified, the event type is not used in determining whether the event should be triggered.
hold-down	(Optional) When a mac-address-table event comes in, the hold-down timer can be set to make the event to wait between 1 and 4294967295 seconds before processing the policy. If not set then the policy is not delayed in being processed.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

Result String

None

Set _cerrno

No

Event_reqinfo For EEM_EVENT_MAT

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u notification %u intf_name %u mac_address {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.
event_severity	The severity of the event.
notification	Notification type--add or delete.
intf_name	The interface name for the address table entry.
mac_address	The mac-address for the address table entry.

event_register_neighbor_discovery

Registers for a neighbor discover event. Use this Tcl command extension to generate an event when a Cisco Discovery Protocol (CDP) or Link Layer Discovery Protocol (LLDP) cache entry or a interface link status changes.

Syntax

```
event_register_neighbor_discovery [tag ?] interface ?
[cdp {add | update | delete | all}]
[lldp {add | update | delete | all}]
[link-event]
[line-event]
[queue_priority {normal | low | high | last}]
[maxrun ?] [nice {0 | 1}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
interface	A regular expression pattern to match against interface names.
cdp	Trigger an event when a matching CDP event occurs. One of the following options should be specified. • add--Trigger events only when a new CDP cache entry is created in the CDP table. • all--Trigger an event when a CDP cache entry is added or deleted from the CDP cache table and when a remote CDP device sends a keepalive to update the CDP cache entry. • delete--trigger events only when a CDP cache entry is deleted from the CDP table. • update--trigger an event when a CDP cache entry is added to the CDP table or when the remote CDP device sends a CDP keepalive to update the CDP cache entry.
lldp	Trigger an event when a matching lldp event occurs. One of the following options should be specified. • add--Trigger events only when a new cdp cache entry is created in the cdp table. • all--Trigger an event when a cdp cache entry is added or deleted from the cdp cache table and when a remote cdp device sends a keepalive to update the cdp cache entry. • delete--trigger events only when a cdp cache entry is deleted from the cdp table. • update--trigger an event when a cdp cache entry is added to the cdp table or when the remote cdp device sends a cdp keepalive to update the cdp cache entry.
line-event	Trigger an event when the interface line protocol status changes.
link-event	Trigger an event when the interface link status changes.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 31536000, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo For EEM_EVENT_NEIGHBOR_DISCOVERY

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u nd_notification {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, at which the event was published to the EEM.

Event Type	Description
event_severity	The severity of the event.
Common Event_Reqinfo	
nd_notification	The type of notification--cdp-add, cdp-update, cdp-delete, lldp-add, lldp-update, lldp-delete, link, line.
nd_intf_linkstatus	The current interface link status, up or down.
nd_intf_linestatus	The current interface line status, down, goingdown, init, testing, up, reset, admindown, deleted.
nd_local_intf_name	The local interface name for the event.
nd_short_local_intf_name	The short name of the local interface for the event.
nd_port_id	The port id as identified by either the cdp or lldp protocol. This is not set for link or line protocol events.
CDP-specific Event_reqinfo	
nd_protocol	Identifies which protocol triggered the event, for CDP it will always be set to cdp.
nd_proto_notif	Identifies which type of protocol event triggered the event, add, update or delete.
nd_proto_new_entry	If set to 1, the event was triggered because the cache entry is new, otherwise it will be set to 0.
nd_cdp_entry_name	The name of the cdp cache entry in the cdp table.
nd_cdp_hold_time	The time remaining until the cdp cache entry expires and is deleted from the cdp table. This time will be reset to some maximum by an update from the cdp neighbor. It is usually set to 0 for new entries.
nd_cdp_mgmt_domain	The CDP VTP management domain.
nd_cdp_platform	The platform name reported by the remote device.
nd_cdp_version	The version of code running on the remote device.
nd_cdp_capabilities_string	The contents of the CDP capabilities field in a string format: Router, Trans-Bridge, Source-Route-Bridge, Switch, Host, IGMP, Repeater, Phone, Remotely-Managed device, CVTA phone port, Two-port Mac Relay or any combination of these separated by commas.
nd_cdp_capabilities_bits	The CDP capabilities bits in a hexadecimal number preceded with 0x.
nd_cdp_capabilities_bit_[0-31]	A series of values that will be set to YES if that bit in the capabilities field is set or NO if it is not set.

Event Type	Description
LLDP-specific Event_reqinfo	
nd_protocol	Identifies which protocol triggered the event, for LLDP it will always be set to lldp.
nd_proto_notif	Identifies which type of protocol event triggered the event, add, update or delete.
nd_proto_new_entry	If set to 1, the event was triggered because the cache entry is new, otherwise it will be set to 0.
nd_lldp_chassis_id	The chassis id field from the LLDP cache entry.
nd_lldp_system_name	The system name from the LLDP cache entry.
nd_lldp_system_description	The system description field from the LLDP cache entry.
nd_lldp_ttl	The LLDP time to live field from the LLDP cache entry.
nd_lldp_port_description	The port description field from the LLDP cache entry.
nd_lldp_system_capabilities_string	The LLDP system capabilities field from the LLDP cache entry. Provided as a string that can contain O, P, B, W, R, T, C, S or any combination of these separated by commas.
nd_lldp_enabled_capabilities_string	The LLDP enabled system capabilities field from the LLDP cache entry. Provided as a string that can contain O, P, B, W, R, T, C, S or any combination of these separated by commas.
nd_lldp_system_capabilities_bits	The LLDP system capabilities bits field from the LLDP cache entry. Provided as a hexadecimal number preceded by 0x.
nd_lldp_enabled_capabilities_bits	The LLDP enabled capabilities bits field from the LLDP cache entry. Provided as a hexadecimal number preceded by 0x.
nd_lldp_capabilities_bits	The LLDP capabilities bits field from the LLDP cache entry. Provided as a hexadecimal number preceded by 0x.
nd_lldp_capabilities_bit_[0-31]	A series of values that will be set to YES if that bit in the capabilities field is set or NO if it is not set.

event_register_nf

Registers for an event when a NetFlow event is triggered by the **event nf**command. Use this Tcl command to publish an event when an NetFlow reaction is triggered..

Syntax

```
event_register_nf [tag ?] monitor_name ? event_type create|update|delete
exit_event_type create|update|delete event1-event4 ? [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
monitor_name	(Mandatory) The name of the NetFlow monitor.
event_type	(Mandatory) The type of event to monitor for the create, update, and delete flow.
exit_event_type	(Mandatory) The event-type (create, delete, update) at which the event is rearmed to be monitored again.
event1- event4	(Mandatory) Specifies the event and its attributes to monitor. Valid values are **event1**, **event2**, **event3**, and **event4**. The subevent keywords can be used alone, together, or in any combination with each other, but each keyword can be used only once.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSS[.MMM] format, where SSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Subevent Syntax

```
field ? rate_interval ? event1 only entry_value ? entry_op eq|ge|gt|le|lt|wc
[exit_value ?] [exit_op eq|ge|gt|le|lt|wc] [exit_rate_interval ? event1 only]
```

Subevent Arguments

field	(Mandatory) Specifies the cache or field attribute to be monitored. One of the following attributes can be specified: • **counter** {**bytes** | **packets**}--Specifies the counter fields. • **datalink** {**dot1q** | **mac**}--Specifies the datalink (layer2) fields. • **flow** {**direction** | **sampler**}--Specifies the flow identifying fields. • **interface** {**input** | **output**}--Specifies the interface fields. • **ipv4** *field-type*-- Specifies the IPv4 fields. • **ipv6** *field-type*-- IPv6 fields • **routing** *routing-attrribute* -- Specifies the routing attributes. • **timestamp sysuptime** {**first | last**}--Specifies the timestamp fields. • **transport** *field-type*-- Specifies the Transport layer fields.
rate_interval	(Mandatory) Specifies the rate interval value in seconds used to calculate the rate. This field is only valid for event1.

entry_value	(Mandatory) Specifies the field or rate value.
entry_op	(Mandatory) Specifies the field operator. The comparison operator valid values are: • **eq** - Equal to • **ge** - Greater than or equal to • **gt** - Greater than • **le** - Less than or equal to • **lt** - Less than • **wc** - Wildcard
exit_value	(Optional) The value at which the event is rearmed to be monitored again.
exit_op	(Optional) The comparison operator used to compare the current event field or rate value with the exit value; if true, event monitoring for this event is reenabled. The comparison operator valid values are: • **eq** - Equal to • **ge** - Greater than or equal to • **gt** - Greater than • **le** - Less than or equal to • **lt** - Less than • **wc** - Wildcard
exit_rate_interval	(Optional) Specifies the exit rate interval value in seconds used to calculate the exit rate value. This field is only valid for event1.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_ID %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u monitor_name %u event1-event4_field %u event1-event4_value
```

Event Type	Description

event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	The type of event to monitor for the create, update, and delete flow.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the NetFlow event.
montior_name	The name of the NetFlow monitor.
event1-event4_field	Specifies the event and its attributes to monitor. Valid values are **event1**, **event2**, **event3**, and **event4**.
event1-event4_value	Specifies the event value and its attributes to monitor. Valid values are **event1**, **event2**, **event3**, and **event4**.

event_register_none

Registers for an event that is triggered by the **event manager run** command. These events are handled by the None event detector that screens for this event.

Syntax

```
event_register_none [tag ?] [sync {yes|no}] [default ?] [queue_priority low|normal|high|last]
 [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
sync	(Optional) A "yes" or a "no" is required to complete this keyword. • If the yes keyword is specified, the policy will run synchronously with the CLI command. • If the no keyword is specified, the policy will run asynchronously with the CLI command.
default	(Optional) The time period during which the CLI event detector waits for the policy to exit (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to run the command. If this argument is not specified, the default time period is set to 30 seconds.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u arg %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
argc **arg1** **arg2** **arg3** **arg4** **arg6** **arg7** **arg8** **arg9** **arg10** **arg11** **arg12** **arg13** **arg14** **arg15**	The parameters that are passed from the XML SOAP command to the script.

event_register_oir

Registers for an online insertion and removal (OIR) event. Use this Tcl command extension to run a policy on the basis of an event raised when a hardware card OIR occurs. These events are handled by the OIR event detector that screens for this event.

Syntax

```
event_register_oir [tag ?] [queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"slot %u event %s"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event ID.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
slot	Slot number for the affected card.
event	Indicates a string, removed or online, that represents either an OIR removal event or an OIR insertion event.

event_register_process

Registers for a process event. Use this Tcl command extension to run a policy on the basis of an event raised when a Cisco IOS Software Modularity process starts or stops. These events are handled by the System Manager event detector that screens for this event. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
event_register_process [tag ?] abort|term|start|user_restart|user_shutdown
[sub_system ?] [version ?] [instance ?] [path ?] [node ?]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
abort	(Mandatory) Abnormal process termination. Process may terminate because of exiting with a nonzero exit status, receiving a kernel-generated signal, or receiving a SIGTERM or SIGKILL signal that is not sent because of user request.
term	(Mandatory) Normal process termination.
start	(Mandatory) Process start.
user_restart	(Mandatory) Process termination due to the process restart request from the CLI command.
user_shutdown	(Mandatory) Process termination due to the process kill request from the CLI command.
sub_system	(Optional) Number assigned to the EEM policy that published the process event. Number is set to 798 because all other numbers are reserved for Cisco use.
version	(Optional) Version number of the process assigned by the version manager. Must be of the form major_number.minor_number.level. If specified, each component of the version number must be an integer between 1 and 4294967295, inclusive.
instance	(Optional) Process instance ID. If specified, this argument must be an integer between 1 and 4294967295, inclusive.

path	(Optional) Process pathname (a regular expression string). If the value of the process-name argument contains embedded blanks, enclose it in double quotation marks. Use path ".*" to match all processes.
node	(Optional) The node name is a string that consists of the word "node" followed by two fields separated by a slash character using the following format: node<slot-number>/<cpu-number> The slot-number is the hardware slot number. The cpu-number is the hardware CPU number. For example, the SP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be specified as node0/0. The RP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be addressed as node0/1. If the node argument is not specified, the default node specification is always the regular expression pattern match of * representing all applicable nodes.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If an optional argument is not specified, the event matches all possible values of the argument. If multiple arguments are specified, the process event will be raised when all the conditions are matched.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"sub_system 0x%x instance %u process_name {%s} path {%s} exit_status 0x%x"
"respawn_count %u last_respawn_sec %ld last_respawn_msec %ld fail_count %u"
"dump_count %u node_name {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
sub_system	Number assigned to the EEM policy that published the application-specific event. Number is set to 798 because all other numbers are reserved for Cisco use.
instance	Process instance ID.
process_name	Process name.
path	Process absolute name including path.
exit_status	Process last exit status.
respawn_count	Number of times that the process was restarted.
last_respawn_sec last_respawn_msec	The calendar time when the last restart occurred.
fail_count	Number of restart attempts of the process that failed. This count will be reset to 0 when the process is successfully restarted.
dump_count	Number of core dumps taken of the process.
node_name	Name of the node that the process is on. The node name is a string that consists of the word "node" followed by two fields separated by a slash character using the following format: **node** *slot-number* / *cpu-number* The slot-number is the hardware slot number. The cpu-number is the hardware CPU number.

event_register_resource

Registers for an Embedded Resource Manager (ERM) event. Use this Tcl command extension to run a policy on the basis of an ERM event report for a specified policy. ERM events are screened by the EEM Resource event detector, allowing an EEM policy to be run when a match occurs for the specified ERM policy.

Syntax

```
event_register_resource policy policy-name [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

policy	(Mandatory) Specifies the use of a policy.
policy-name	(Mandatory) Name of an ERM policy.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"owner_id %lld user_id %lld" time_sent %llu dampen_time %d notify_data_flags %u"
"level {%s} direction {%s} configured_threshold %u current_value %u"
"policy_violation_flag {%s} policy_id %d"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
owner_id	The Embedded Resource Manager (ERM) owner ID.
user_id	The ERM user ID.
time_sent	The ERM event time, in nanoseconds.
dampen_time	The ERM dampen time, in nanoseconds.
notify_data_flags	The ERM notify data flag.
level	The ERM event level. The four event levels are normal, minor, major, and critical.
direction	The ERM event direction. The event direction can be one of the following: up, down, or no change.
configured_threshold	The configured ERM threshold.
current_value	The current value reported by ERM.
policy_violation_flag	The ERM policy violation flag; either false or true.
policy_id	The ERM policy ID.

event_register_rf

Registers for a Redundancy Facility (RF) event. Use this Tcl command extension to run a policy when an RF progression or status event notification occurs.

Syntax

```
event_register_rf [tag ?] event ?
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
event	(Mandatory) Name of the RF progression or status event. Valid values are: • RF_PROG_ACTIVE • RF_PROG_ACTIVE_DRAIN • RF_PROG_ACTIVE_FAST = 200 • RF_PROG_ACTIVE_PRECONFIG • RF_PROG_ACTIVE_POSTCONFIG • RF_PROG_EXTRALOAD • RF_PROG_HANDBACK • RF_PROG_INITIALIZATION • RF_PROG_PLATFORM_SYNC • RF_PROG_STANDBY_BULK • RF_PROG_STANDBY_COLD • RF_PROG_STANDBY_CONFIG • RF_PROG_STANDBY_FILESYS • RF_PROG_STANDBY_HOT • RF_PROG_STANDBY_OIR_SYNC_DONE • RF_REGISTRATION_STATUS • RF_STATUS_MAINTENANCE_ENABLE • RF_STATUS_MANUAL_SWACT • RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE • RF_STATUS_PEER_COMM • RF_STATUS_PEER_PRESENCE • RF_STATUS_REDUNDANCY_MODE_CHANGE • RF_STATUS_SWACT_INHIBIT

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event	RF progression or status event notification that caused this event to be published.

event_register_routing

Registers for an event that is triggered by the **event routing** command. These events are handled by the routing event detector to publish an event when route entries change in Routing Information Base (RIB) infrastructure. Use this Tcl command extension to run a routing policy for this script. The network IP address for the route to be monitored must be specified.

Syntax

```
event_register_routing [tag ?] network ? length [ge|le|ne] [type add|remove|modify|all]
[protocol ?] [queue_priority normal|low|high|last] [maxrun ?] [nice {0 | 1}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
network	Specifies the network IP address. The network number can be any valid IP address or prefix.
length	Specifies the length of the network mask in bits. The bit mask can be a number from 0 to 32. • **ge** --(Optional) Specifies the minimum prefix length to be matched. The **ge** keyword represents greater than or equal to operator. • **le** --(Optional) Specifies the maximum prefix length to be matched. The **le** keyword represents the less than or equal to operator. • **ne** --(Optional) Specifies the prefix length not to be matched. The **ne** keyword represents not equal to operator. When **ge**, **le** and **ne** keywords are not configured, an exact match of network length is processed.
type	(Optional) Specifies the desired policy trigger. The type options are **add**, **remove**, **modify**, and **all**. The default is **all**.
protocol	(Optional) Specifies the protocol value for the network being monitored. One of the following protocols can be used: **all**, **bgp**, **connected**, **eigrp**, **isis**, **iso-igrp**, **mobile**, **odr**, **ospf**, **rip**, and **static**. The default is **all**.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u" "event_severity {%s} %u network %u mask %u protocol %u lastgateway %u distance %u" "time_sec %u time_msec %u metric %u lastinterface %u"

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
network	The network prefix in IP address format
mask	The network mask in IP address format
protocol	Type of network protocol.
type	Type of event to add, remove or modify.
lastgateway	The last known gateway.
distance	The administrative distance.
time_sec time_msec	Time of event in seconds and milliseconds, when the event was published to the EEM.
metric	Path metric.
lastinterface	The last known interface.

event_register_rpc

Registers for an event that is triggered by the EEM SSH Remote Procedure Call (RPC) command. These events are handled by the RPC event detector that screens for this event. Use this Tcl command extension to run a RPC policy for this script.

Syntax

```
event_register_rpc [queue_priority {normal | low | high | last}] [maxrun <sec.msec>] [nice
 {0 | 1}] [default <sec.msec>]
```

Arguments

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.
default	(Optional) The time period during which the CLI event detector waits for the policy to exit (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to run the command. If this argument is not specified, the default time period is set to 30 seconds.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
arg %u"
```

Event Type	Description

event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
argc **arg0** **arg1** **arg2** **arg3** **arg4** **arg6** **arg7** **arg8** **arg9** **arg10** **arg11** **arg12** **arg13** **arg14**	The parameters that are passed from the XML SOAP command to the script.

event_register_snmp

Registers for a Simple Network Management Protocol (SNMP) statistics event. Use this Tcl command extension to run a policy when a given counter specified by an SNMP object ID (oid) crosses a defined threshold.

Syntax

```
event_register_snmp [tag ?] oid ? get_type exact|next
entry_op gt|ge|eq|ne|lt|le entry_val ?
entry_type value|increment|rate
[exit_comb or|and]
[exit_op gt|ge|eq|ne|lt|le] [exit_val ?]
[exit_type value|increment|rate]
[exit_time ?] poll_interval ? [average_factor ?]
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
oid	(Mandatory) OID number of data element in SNMP dot notation (for example, 1.3.6.1.2.1.2.1.0). The types of OIDs allowed are: • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
entry_op	(Mandatory) Entry comparison operator used to compare the current OID data value with the entry value; if true, an event will be raised and event monitoring will be disabled until exit criteria are met.
get_type	(Mandatory) Type of SNMP get operation that needs to be applied to the OID specified. If the get_type argument is "exact," the value of the specified OID is retrieved; if the get_type argument is "next," the value of the lexicographical successor to the specified OID is retrieved.
entry_val	(Mandatory) Value with which the current oid data value should be compared to decide if the SNMP event should be raised.
entry-type	Specifies a type of operation to be applied to the object ID specified by the entry-val argument. Value is defined as the actual value of the entry-val argument. Increment uses the entry-val field as an incremental difference and the entry-valis compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_comb	(Optional) Exit combination operator used to indicate the combination of exit condition tests required to decide if the exit criteria are met so that the event monitoring can be reenabled. If it is "and," both exit value and exit time tests must be passed to meet the exit criteria. If it is "or," either exit value or exit time tests can be passed to meet the exit criteria. When exit_comb is "and," exit_op, and exit_val (exit_time) must exist. When exit_comb is "or," (exit_op and exit_val) or (exit_time) must exist.
exit_op	(Optional) Exit comparison operator used to compare the current oid data value with the exit value; if true, event monitoring for this event will be reenabled.

exit_val	(Optional) Value with which the current oid data value should be compared to decide if the exit criteria are met.
exit-type	(Optional) Specifies a type of operation to be applied to the object ID specified by the exit-val argument. If not specified, the value is assumed. Value is defined as the actual value of the exit-val argument. Increment uses the exit-val field as an incremental difference and the exit-val is compared with the difference between the current counter value and the value when the event was last triggered (or the first polled sample if this is a new event). A negative value checks the incremental difference for a counter that is decreasing. Rate is defined as the average rate of change over a period of time. The time period is the average-factor value multiplied by the poll-interval value. At each poll interval the difference between the current sample and the previous sample is taken and recorded as an absolute value. An average of the previous average-factor value samples is taken to be the rate of change.
exit_time	(Optional) Number of POSIX timer units after an event is raised when event monitoring will be enabled again. Specified in SSSSSSSSSS[.MMM] format where SSSSSSSSSS must be an integer number representing seconds between 0 and 4294967295, inclusive. MMM represents milliseconds and must be an integer number between 0 and 999.
poll_interval	(Mandatory) Interval between consecutive polls in POSIX timer units. Currently the interval is forced to be at least 1 second (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).
average-factor	(Optional) Number in the range from 1 to 64 used to calculate the period used for rate-based calculations. The average-factor value is multiplied by the poll-interval value to derive the period in milliseconds. The minimum average factor value is 1.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.

maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event_severity {%s} oid {%s} val {%s} delta_val {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	SNMP event severity, which can be one of the following values: • normal • minor • major
oid	Object ID of data element, in SNMP dot notation.
val	Value of the data element.
delta_val	Delta value between the value of the policies.

event_register_snmp_notification

Registers for a Simple Network Management Protocol (SNMP) notification trap event. Use this Tcl command extension to run a policy when an SNMP trap with the specified SNMP object ID (oid) is encountered on a specific interface or address. The **snmp-server manager** CLI command must be enabled for the SNMP notifications to work using Tcl policies.

Syntax

```
event_register_snmp_notification [tag ?] oid ? oid_val ?
op {gt|ge|eq|ne|lt|le}
[maxrun ?]
[src_ip_address ?]
[dest_ip_address ?]
[queue_priority {normal|low|high|last}]
[maxrun ?]
[nice {0|1}]
[default ?]
[direction {incoming|outgoing}]
[msg_op {drop|send}]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
oid	(Mandatory) OID number of the data element in SNMP dot notation (for example, 1.3.6.1.2.1.2.1.0). If the specified OID ends with a dot (.), then all OIDs that start with the OID number before the dot are matched. The types of OIDs allowed are: • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
oid_val	(Mandatory) OID value with which the current OID data value should be compared to decide if the SNMP event should be raised.
op	(Mandatory) Comparison operator used to compare the current OID data value with the SNMP Protocol Data Unit (PDU) OID data value; if this is true, an event is raised.
maxrun	(Optional) Maximum run time of the script (specified in ssssssss[.mmm] format, where ssssssss must be an integer representing seconds between 0 and 31536000, inclusive, and where mmm must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

src_ip_address	(Optional) Source IP address where the SNMP notification trap originates. The default is all; it is set to receive SNMP notification traps from all IP addresses.
dest_ip_address	(Optional) Destination IP address where the SNMP notification trap is sent. The default is all; it is set to receive SNMP traps from all destination IP addresses.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the queue_priority_last argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
default	(Optional) Specifies the time period in seconds during which the snmp notification event detector waits for the policy to exit. The time period is specified in sssssssss[.mmm] format, where sssssssss must be an integer representing seconds between 0 and 4294967295 and mmm must be an integer representing milliseconds between 0 and 999.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.
direction	(Optional) The direction of the incoming or outgoing SNMP trap or inform PDU to filter. The default value is incoming.
msg_op	(Optional) The action to be taken on the SNMP PDU (drop it or send it) once the event is triggered. The default value is send.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u
event_severity {%s}" "oid {%s} oid_val {%s} src_ip_addr {%s} dest_ip_addr {%s} x_x_x_x_x
```

```
(varbinds) {%s} trunc_vb_buf {%s} trap_oid {%s} enterprise_oid {%s} generic_trap %u
specific_trap %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
oid	An user specified object ID.
oid_val	An user specified object ID value.
src_ip_addr	The source IP address of the SNMP protocol data unit (PDU).
dest_ip_addr	The destination IP address of the SNMP PDU.
x_x_x_x_x (varbinds)	The SNMP PDU varbind information.
trap_oid	Indicates the trap OID value.
enterprise_oid	Indicates the enterprise OID value.
generic_trap	Indicates one of a number of generic trap types. There are seven generic trap numbers zero to six.
specific_trap	Indicates one of a number of specific trap codes.

event_register_snmp_object

Registers for a Simple Network Management Protocol (SNMP) object event. Use this Tcl command extension to replace the value when an SNMP with the specified SNMP-object ID (OID) is encountered on a specific interface or address.

Syntax

```
event_register_snmp_object oid ?
type {int|uint|counter|counter64|gauge|ipv4||oid|string}
sync {yes|no}
skip {yes|no}
[istable {yes|no}]
[default ?]
[queue_priority {normal|low|high|last}]
[maxrun ?]
[nice {0|1}]
```

Arguments

oid	(Mandatory) OID number of the data element in SNMP dot notation (for example, 1.3.6.1.2.1.2.1.0). If the specified OID ends with a dot (.), then all OIDs that start with the OID number before the dot are matched. The types of OIDs allowed are: • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
type	(Mandatory) OID value type.
sync	(Mandatory) A "yes" means that the EEM policy will be notified. If the applet set_exit_status or Tcl return value is 0, then SNMP will handle the request. If the return value is 1, SNMP will use the value provided by the policy for the get request and will not process the set request. A "no" means that EEM will not be notified and SNMP will handle the request. Only one OID can be associated with a synchronous policy. However, multiple synchronous policies can be registered for the same OID.
skip	Mandatory if the sync argument is "no" and should not exist if the sync argument is "yes." If the skip argument is "yes," it means that SNMP will handle the request. If the skip argument is "no," it means that SNMP will act as if the object does not exist.
istable	(Optional) A value of "no" means the OID is scalar object, and "yes" means the OID is table object.
default	(Optional) The time period during which the SNMP Object event detector waits for the policy to exit (specified in sssssssss[.mmm] format, where sssssssss must be an integer representing seconds between 0 and 4294967295, inclusive, and where mmm must be an integer representing milliseconds between 0 and 999). If the default time period expires before the policy exits, the default action will be executed. The default action is to process the set or get request normally by SNMP subsystem. If this argument is not specified, the default time period is set to 30 seconds.
maxrun	(Optional) Maximum run time of the script (specified in ssssssss[.mmm] format, where ssssssss must be an integer representing seconds between 0 and 31536000, inclusive, and where mmm must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the queue_priority_last argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u
event_severity {%s}" "oid {%s} request {%s} request_type {%s} value %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
event_severity	The severity of the event.
oid	The ID of the SNMP object in the received get or set request.

Event Type	Description
request	The get or set request type.
request_type	The type of request (exact or next).
value	For set requests only. The value to set the object to.

event_register_syslog

Registers for a syslog event. Use this Tcl command extension to trigger a policy when a syslog message of a specific pattern is logged after a certain number of occurrences during a certain period of time.

Syntax

```
event_register_syslog [tag ?] [occurs ?] [period ?] pattern ?
[priority all|emergencies|alerts|critical|errors|warnings|notifications|
informational|debugging|0|1|2|3|4|5|6|7]
[queue_priority low|normal|high|last]
[severity_fatal] [severity_critical] [severity_major]
[severity_minor] [severity_warning] [severity_notification]
[severity_normal] [severity_debugging]
[maxrun ?] [nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
occurs	(Optional) Number of occurrences before the event is raised; if not specified, the event is raised on the first occurrence. If specified, the value must be greater than 0.
period	(Optional) Time interval, in seconds and milliseconds, during which the one or more occurrences must take place in order to raise an event (specified in SSSSSSSSSS[.MMM] format where SSSSSSSSSS must be an integer number representing seconds between 0 and 4294967295, inclusive, and where MMM represents milliseconds and must be an integer number between 0 and 999). If this argument is not specified, no period check is applied.
pattern	(Mandatory) A regular expression used to perform syslog message pattern match. This argument is what the policy uses to identify the logged syslog message.
priority	(Optional) The message priority to be screened. If this argument is specified, only messages that are at the specified logging priority level, or lower, are screened. If this argument is not specified, the default priority is 0.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.
severity_xxx	(Optional) The event severity to be screened. If this argument is specified, only messages that are at the specified severity level are screened. See the table titled "Severity Level Mapping For Syslog Events" for the severity level mapping for syslog events.

If multiple conditions are specified, the syslog event will be raised when all the conditions are matched.

Table 1: Severity Level Mapping For Syslog Events

Severity Keyword	Syslog Priority	Description
severity_fatal	LOG_EMERG (0)	System is unusable.
severity_critical	LOG_ALERT (1)	Critical conditions, immediate attention required.
severity_major	LOG_CRIT (2)	Major conditions.
severity_minor	LOG_ERR (3)	Minor conditions.
severity_warning	LOG_WARNING (4)	Warning conditions.
severity_notification	LOG_NOTICE (5)	Basic notification, informational messages.
severity_normal	LOG_INFO (6)	Normal event, indicates returning to a normal state.
severity_debugging	LOG_DEBUG (7)	Debugging messages.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"msg {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
msg	The last syslog message that matches the pattern.

event_register_timer

Creates a timer and registers for a timer event as both a publisher and a subscriber. Use this Tcl command extension when there is a need to trigger a policy that is time specific or timer based. This event timer is both an event publisher and a subscriber. The publisher part indicates the conditions under which the named timer is to go off. The subscriber part identifies the name of the timer to which the event is subscribing.



Note Both the CRON and absolute time specifications work on local time.

Syntax

```
event_register_timer [tag ?] watchdog|countdown|absolute|cron
[name ?] [cron_entry ?]
[time ?]
[queue_priority low|normal|high|last] [maxrun ?]
[nice 0|1]
```

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
watchdog	(Mandatory) Watchdog timer.
countdown	(Mandatory) Countdown timer.
absolute	(Mandatory) Absolute timer.
cron	(Mandatory) CRON timer.
name	(Optional) Name of the timer.
cron_entry	(Optional) Must be specified if the CRON timer type is specified. Must not be specified if any other timer type is specified. A cron_entry is a partial UNIX crontab entry (the first five fields) as used with the UNIX CRON daemon. A cron_entry specification consists of a text string with five fields. The fields are separated by spaces. The fields represent the time and date when CRON timer events will be triggered. The fields are described in the table titled "Time and Date When CRON Events Will Be Triggered." Ranges of numbers are allowed. Ranges are two numbers separated with a hyphen. The specified range is inclusive. For example, 8-11 for an hour entry specifies execution at hours 8, 9, 10, and 11. A field may be an asterisk (*), which always stands for "first-last." Lists are allowed. A list is a set of numbers (or ranges) separated by commas. Examples: "1,2,5,9" and "0-4,8-12". Step values can be used in conjunction with ranges. Following a range with "/<number>" specifies skips of the number's value through the range. For example, "0-23/2" can be used in the hour field to specify an event that is triggered every other hour. Steps are also permitted after an asterisk, so if you want to say "every two hours", use "*/2". Names can also be used for the month and the day of week fields. Use the first three letters of the particular day or month (case does not matter). Ranges or lists of names are not allowed. The day on which a timer event is triggered can be specified by two fields: day of month and day of week. If both fields are restricted (that is, are not *), an event will be triggered when either field matches the current time. For example, "30 4 1,15 * 5" would cause an event to be triggered at 4:30 a.m. on the 1st and 15th of each month, plus every Friday. Instead of the first five fields, one of seven special strings may appear. These seven special strings are described in the table titled "Special Strings for cron_entry." Example 1: "0 0 1,15 * 1" would trigger an event at midnight on the 1st and 15th of each month, as well as on every Monday. To specify days by only one field, the other field should be set to *; "0 0 * * 1" would trigger an event at midnight only on Mondays. Example 2: "15 16 1 * *" would trigger an event at 4:15 p.m. on the first day of each month. Example 3: "0 12 * * 1-5" would trigger an event at noon on Monday through Friday of each week. Example 4: "@weekly" would trigger an event at midnight once a week on Sunday.

time	(Optional) Must be specified if a timer type other than CRON is specified. Must not be specified if the CRON timer type is specified. For watchdog and countdown timers, the number of seconds and milliseconds until the timer expires; for the absolute timer, the calendar time of the expiration time. Time is specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999. An absolute expiration date is the number of seconds and milliseconds since January 1, 1970. If the date specified has already passed, the timer expires immediately.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Table 2: Time and Date When CRON Events Will Be Triggered

Field	Allowed Values
minute	0-59
hour	0-23
day of month	1-31
month	1-12 (or names, see below)
day of week	0-7 (0 or 7 is Sun, or names; see the table titled "Special Strings for cron_entry")

Table 3: Special Strings for cron_entry

String	Meaning
@yearly	Trigger once a year, "0 0 1 1 *".
@annually	Same as @yearly.
@monthly	Trigger once a month, "0 0 1 * *".
@weekly	Trigger once a week, "0 0 * * 0".
@daily	Trigger once a day, "0 0 * * *".
@midnight	Same as @daily.
@hourly	Trigger once an hour, "0 * * * *".

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"timer_type %s timer_time_sec %ld timer_time_msec %ld"
"timer_remain_sec %ld timer_remain_msec %ld"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
timer_type	Type of the timer. Can be one of the following: • watchdog • countdown • absolute
timer_time_sec timer_time_msec	Time when the timer expired.

Event Type	Description
timer_remain_sec **timer_remain_msec**	The remaining time before the next expiration.

See Also

event_register_timer_subscriber

event_register_timer_subscriber

Registers for a timer event as a subscriber. Use this Tcl command extension to identify the name of the timer to which the event timer, as a subscriber, wants to subscribe. The event timer depends on another policy or another process to actually manipulate the timer. For example, let policyB act as a timer subscriber policy, but policyA (although it does not need to be a timer policy) uses register_timer, timer_arm, or timer_cancel Tcl command extensions to manipulate the timer referenced in policyB.

Syntax

```
event_register_timer_subscriber watchdog|countdown|absolute|cron
name ? [queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

Arguments

watchdog	(Mandatory) Watchdog timer.
countdown	(Mandatory) Countdown timer.
absolute	(Mandatory) Absolute timer.
cron	(Mandatory) CRON timer.
name	(Mandatory) Name of the timer.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.



Note An EEM policy that registers for a timer event or a counter event can act as both publisher and subscriber.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"timer_type %s timer_time_sec %ld timer_time_msec %ld"
"timer_remain_sec %ld timer_remain_msec %ld"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.

Event Type	Description
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
timer_type	Type of the timer. Can be one of the following: • watchdog • countdown • absolute
timer_time_sec timer_time_msec	Time when the timer expired.
timer_remain_sec **timer_remain_msec**	The remaining time before the next expiration.

See Also

event_register_timer

event_register_track

Registers for a report event from the Cisco IOS Object Tracking subsystem. Use this Tcl command extension to trigger a policy on the basis of a Cisco IOS Object Tracking subsystem report for a specified object number.

Syntax

```
event_register_track ? [tag ?] [state up|down|any] [queue_priority low|normal|high|last]
[maxrun ?]
[nice 0|1]
```

Arguments

? (represents a number)	(Mandatory) Tracked object number in the range from 1 to 500, inclusive.
tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
state	(Optional) Specifies that the tracked object transition will cause an event to be raised. If **up** is specified, an event will be raised when the tracked object transitions from a down state to an up state. If **down** is specified, an event will be raised when the tracked object transitions from an up state to a down state. If **any** is specified, an event will be raised when the tracked object transitions to or from any state.

queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.
maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

If an optional argument is not specified, the event matches all possible values of the argument.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"track_number {%u} track_state {%s}"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event ID.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.

Event Type	Description
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
track_number	Number of the tracked object that caused the event to be triggered.
track_state	State of the tracked object when the event was triggered; valid states are up or down.

event_register_wdsysmon

Registers for a Watchdog system monitor event. Use this Tcl command extension to register for a composite event which is a combination of several subevents or conditions. For example, you can use this command to register for the combination of conditions wherein the CPU usage of a certain process is over 80 percent and the memory used by the process is greater than 50 percent of its initial allocation. This Tcl command extension is supported only in Software Modularity images.

Syntax

```
event_register_wdsysmon [tag ?] [timewin ?]
[sub12_op and|or|andnot]
[sub23_op and|or|andnot]
[sub34_op and|or|andnot]
[sub1 subevent-description]
[sub2 subevent-description]
[sub3 subevent-description]
[sub4 subevent-description] [node ?]
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

Each argument is position independent.



Note Operator definitions: and (logical and operation), or (logical or operation), andnot (logical and not operation). For example, "sub12_op and" is defined as raise an event when subevent 1 and subevent 2 are true; "sub23_op or" is defined as raise an event when the condition specified in sub12_op is true or subevent 3 is true. The logic can be diagrammed using: if (((sub1 sub12_op sub2) sub23_op sub3) sub34_op sub4) is TRUE, raise event

Arguments

tag	(Optional) String identifying a tag that can be used with the trigger Tcl command extension to support multiple event statements within a Tcl script.
timewin	(Optional) Time window within which all of the subevents have to occur in order for an event to be generated (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999).

sub12_op	(Optional) Combination operator for comparison between subevent 1 and subevent 2.
sub23_op	(Optional) Combination operator for comparison between subevent 1 and 2 and subevent 3.
sub34_op	(Optional) Combination operator for comparison between subevent 1 and 2 and subevent 3 and subevent 4.
sub1	(Optional) Indicates that subevent 1 is specified.
subevent-description	(Optional) Syntax for the subevent.
sub2	(Optional) Indicates that subevent 2 is specified.
sub3	(Optional) Indicates that subevent 3 is specified.
sub4	(Optional) Indicates that subevent 4 is specified.
node	(Optional) The node name to be monitored for deadlock conditions is a string that consists of the word "node" followed by two fields separated by a slash character using the following format: node<slot-number>/<cpu-number> The slot-number is the hardware slot number. The cpu-number is the hardware CPU number. For example, the SP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be specified as node0/0. The RP CPU in a Supervisor card on a Cisco Catalyst 6500 series switch located in slot 0 would be addressed as node0/1. If the node argument is not specified, the default node specification is the local node on which the registration is done.
queue_priority	(Optional) Priority level at which the script will be queued: • queue_priority low--Specifies that the script is to be queued at the lowest of the three priority levels. • queue_priority normal--Specifies that the script is to be queued at a priority level greater than low priority but less than high priority. • queue_priority high--Specifies that the script is to be queued at the highest of the three priority levels. • queue_priority last--Specifies that the script is to be queued at the lowest priority level. If more than one script is registered with the "queue_priority_last" argument set, these scripts will execute in the order in which the events are published. **Note** The queue_priority argument specifies the queuing priority, but not the execution priority, of the script being registered. If this argument is not specified, the default queuing priority is normal.

maxrun	(Optional) Maximum run time of the script (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the default 20-second run-time limit is used.
nice	(Optional) Policy run-time priority setting. When the nice argument is set to 1, the policy is run at a run-time priority that is less than the default priority. The default value is 0.

Subevents

The syntax of subevent descriptions can be one of seven cases.

For arguments in subevent description, the following constraints apply on the value of number arguments:

- For dispatch_mgr, val must be an integer between 0 and 4294967295, inclusive.
- For cpu_proc and cpu_tot, val must be an integer between 0 and 100, inclusive.
- For mem_proc, mem_tot_avail, and mem_tot_used, if is_percent is FALSE, val must be an integer between 0 and 4294967295, inclusive.

```
1. deadlock procname ?
```

Arguments

procname	(Mandatory) A regular expression that specifies the process name that you wish to monitor for deadlock conditions. This subevent will ignore the time window even if it is given.

```
2. dispatch_mgr [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

Arguments

procname	(Optional) A regular expression that specifies the process name that you wish to monitor for dispatch_manager status.
op	(Optional) Comparison operator used to compare the collected number of events with the specified value; if true, an event will be raised.
val	(Optional) The value with which the number of events that have occurred should be compared.
period	(Optional) The time period for the number of events that have occurred (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
3. cpu_proc [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

Arguments

procname	(Optional) A regular expression that specifies the process name that you wish to monitor for CPU utilization conditions.
op	(Optional) Comparison operator used to compare the collected CPU usage sample percentage with the specified percentage value; if true, an event will be raised.
val	(Optional) The percentage value with which the average CPU usage during the sample period should be compared.
period	(Optional) The time period for averaging the collection of samples (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
4. cpu_tot [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

Arguments

op	(Optional) Comparison operator used to compare the collected total system CPU usage sample percentage with the specified percentage value; if true, an event will be raised.
val	(Optional) The percentage value with which the average CPU usage during the sample period should be compared.
period	(Optional) The time period for averaging the collection of samples (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
5. mem_proc [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

Arguments

procname	(Optional) A regular expression that specifies the process name that you wish to monitor for memory usage.
op	(Optional) Comparison operator used to compare the collected memory used with the specified value; if true, an event will be raised.
val	(Optional) A percentage or an absolute value specified in kilobytes. A percentage represents the difference between the oldest sample in the specified time period and the latest sample. If memory usage has increased from 150 KB to 300 KB within the time period, the percentage increase is 100. This is the value with which the measured value should be compared.
is_percent	(Optional) If TRUE, the percentage value is collected and compared. Otherwise, the absolute value is collected and compared.

period	(Optional) If is_percent is set to TRUE, the time period for the percentage to be computed. Otherwise, the time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
6. mem_tot_avail [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

Arguments

op	(Optional) Comparison operator used to compare the collected available memory with the specified value; if true, an event will be raised.
val	(Optional) A percentage or an absolute value specified in kilobytes. A percentage represents the difference between the oldest sample in the specified time period and the latest sample. If available memory usage has decreased from 300 KB to 150 KB within the time period, the percentage decrease is 50. This is the value with which the measured value should be compared.
is_percent	(Optional) If TRUE, the percentage value is collected and compared. Otherwise, the absolute value is collected and compared.
period	(Optional) If is_percent is set to TRUE, the time period for the percentage to be computed. Otherwise, the time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used.

```
7. mem_tot_used [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

Arguments

op	(Optional) Comparison operator used to compare the collected used memory with the specified value; if true, an event will be raised.
val	(Optional) A percentage or an absolute value specified in kilobytes. A percentage represents the difference between the oldest sample in the specified time period and the latest sample. If memory usage has increased from 150 KB to 300 KB within the time period, the percentage increase is 100. This is the value with which the measured value should be compared.
is_percent	(Optional) If TRUE, the percentage value is collected and compared. Otherwise, the absolute value is collected and compared.

period	(Optional) If is_percent is set to TRUE, the time period for the percentage to be computed. Otherwise, the time period for the collection samples to be averaged (specified in SSSSSSSSSS[.MMM] format, where SSSSSSSSSS must be an integer representing seconds between 0 and 4294967295, inclusive, and where MMM must be an integer representing milliseconds between 0 and 999). If this argument is not specified, the most recent sample is used. **Note** This argument is mandatory if is_percent is set to TRUE; otherwise, it is optional.

Result String

None

Set _cerrno

No

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"num_subs %u"
```

Event Type	Description
event_id	Unique number that indicates the ID for this published event. Multiple policies may be run for the same event, and each policy will have the same event_id.
event_type	Type of event.
event_type_string	An ASCII string that represents the name of the event for this event type.
event_pub_sec event_pub_msec	The time, in seconds and milliseconds, when the event was published to the EEM.
num_subs	Subevent number.

Where the subevent info string is for a deadlock subevent:

```
"{type %s num_entries %u entries {entry 1, entry 2, ...}}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
num_entries	Number of processes and threads in the deadlock.
entries	Information of processes and threads in the deadlock.

Where each entry is:

```
"{node {%s} procname {%s} pid %u tid %u state %s b_node %s b_procname %s b_pid %u
b_tid %u}"
```

Assume that the entry describes the scenario in which Process A thread m is blocked on process B thread n:

Subevent Type	Description
node	Name of the node that process A thread m is on.
procname	Name of process A.
pid	Process ID of process A.
tid	Thread ID of process A thread m.
state	Thread state of process A thread m. Can be one of the following: • STATE_CONDVAR • STATE_DEAD • STATE_INTR • STATE_JOIN • STATE_MUTEX • STATE_NANOSLEEP • STATE_READY • STATE_RECEIVE • STATE_REPLY • STATE_RUNNING • STATE_SEM • STATE_SEND • STATE_SIGSUSPEND • STATE_SIGWAITINFO • STATE_STACK • STATE_STOPPED • STATE_WAITPAGE • STATE_WAITTHREAD
b_node	Name of the node that process B thread is on.
b_procname	Name of process B.
b_pid	Process ID of process B.

Subevent Type	Description
b_tid	Thread ID of process B thread n; 0 means that process A thread m is blocked on all threads of process B.

For dispatch_mgr Subevent

```
"{type %s node {%s} procname {%s} pid %u value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node that the POSIX process is on.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent. **Note** The three fields above describe the owner process of this dispatch manager.
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the number of events processed by the dispatch manager is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the total number of events processed by this dispatch manager is in the given time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

For cpu_proc Subevent

```
"{type %s node {%s} procname {%s} pid %u value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node that the POSIX process is on.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent. **Note** The three fields above describe the process whose CPU utilization is being monitored.

Subevent Type	Description
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the process CPU utilization is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged process CPU utilization is in the given time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

For cpu_tot Subevent

```
"{type %s node {%s} value %u sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node on which the total CPU utilization is being monitored.
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the total CPU utilization is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged total CPU utilization is in the given time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

For mem_proc Subevent

```
"{type %s node {%s} procname {%s} pid %u is_percent %s value %u diff %d sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node that the POSIX process is on.
procname	POSIX process name for this subevent.
pid	POSIX process ID for this subevent. **Note** The three fields above describe the process whose memory usage is being monitored.

Subevent Type	Description
is_percent	Can be either TRUE or FALSE. TRUE means that the value is a percentage value; FALSE means that the value is an absolute value (may be an averaged value).
value	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the process used memory is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged process used memory utilization is in the given time window.
Subevent Type	Description
diff	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **diff** is the percentage difference between the first process used memory sample ever collected and the latest process used memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **diff** is the percentage difference between the oldest and latest process used memory utilization in the specified time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is FALSE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **value** is the process used memory in the latest sample.
- **diff** is 0.
- **sec** and **msec** are both 0.

If the **is_percent** argument is FALSE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **value** is the averaged process used memory sample value in the specified time window.
- **diff** is 0.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is TRUE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **value** is 0.
- **diff** is the percentage difference between the oldest and latest process used memory samples in the specified time window.
- **sec** and **msec** are the actual time difference between the time stamps of the oldest and latest process used memory samples in this time window.

If the **is_percent** argument is TRUE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **value** is 0.
- **diff** is the percentage difference between the first process used memory sample ever collected and the latest process used memory sample.
- **sec** and **msec** are the actual time difference between the time stamps of the first process used memory sample ever collected and the latest process used memory sample.

For mem_tot_avail Subevent

```
"{type %s node {%s} is_percent %s used %u avail %u diff %d sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node for which the total available memory is being monitored.
is_percent	Can be either TRUE or FALSE. TRUE means that the value is a percentage value; FALSE means that the value is an absolute value (may be an averaged value).
used	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the total used memory is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged total used memory utilization is in the given time window.
avail	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **avail** is in the latest total available memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **avail** is the total available memory utilization in the specified time window.
diff	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **diff** is the percentage difference between the first total available memory sample ever collected and the latest total available memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **diff** is the percentage difference between the oldest and latest total available memory utilization in the specified time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, they are the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is FALSE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is the total used memory in the latest sample.
- **avail** is the total available memory in the latest sample.
- **diff** is 0.

- **sec** and **msec** are both 0.

If the **is_percent** argument is FALSE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is 0.
- **avail** is the averaged total available memory sample value in the specified time window.
- **diff** is 0.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total available memory samples in this time window.

If the **is_percent** argument is TRUE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.
- **diff** is the percentage difference between the oldest and latest total available memory samples in the specified time window.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total available memory samples in this time window.

If the **is_percent** argument is TRUE, and the **sec** and **msec** arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.
- **diff** is the percentage difference between the first total available memory sample ever collected and the latest total available memory sample.
- **sec** and **msec** are the actual time difference between the time stamps of the first total available memory sample ever collected and the latest total available memory sample.

For mem_tot_used Subevent

```
"{type %s node {%s} is_percent %s used %u avail %u diff %d sec %ld msec %ld}"
```

Subevent Type	Description
type	Type of wdsysmon subevent.
node	Name of the node for which the total used memory is being monitored.
is_percent	Can be either TRUE or FALSE. TRUE means that the value is a percentage value; FALSE means that the value is an absolute value (may be an averaged value).

Subevent Type	Description
used	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the total used memory is in the latest sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the averaged total used memory utilization is in the given time window.
avail	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **avail** is in the latest total used memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **avail** is the total used memory utilization in the specified time window.
diff	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, the **diff** is the percentage difference between the first total used memory sample ever collected and the latest total used memory sample. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **diff** is the percentage difference between the oldest and latest total used memory utilization in the specified time window.
sec msec	If the **sec** and **msec** variables are specified as 0 or are unspecified in the event registration Tcl command extension, they are both 0. If a time window is specified and is greater than zero in the event registration Tcl command extension, the **sec** and **msec** variables are the actual time difference between the time stamps of the oldest and latest samples in this time window.

If the **is_percent** argument is FALSE, and the sec and msec arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is the total used memory in the latest sample,
- **avail** is the total available memory in the latest sample,
- **diff** is 0,
- **sec** and **msec** are both 0,

If the **is_percent** argument is FALSE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is the averaged total used memory sample value in the specified time window,
- **avail** is 0,
- **diff** is 0,
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total used memory samples in this time window,

If the **is_percent** argument is TRUE, and a time window is specified as greater than zero in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.

- **diff** is the percentage difference between the oldest and latest total used memory samples in the specified time window.
- **sec** and **msec** are both the actual time difference between the time stamps of the oldest and latest total used memory samples in this time window.

If the **is_percent** argument is TRUE, and the sec and msec arguments are specified as 0 or are unspecified in the event registration Tcl command extension:

- **used** is 0.
- **avail** is 0.
- **diff** is the percentage difference between the first total used memory sample ever collected and the latest total used memory sample.
- **sec** and **msec** are the actual time difference between the time stamps of the first total used memory sample ever collected and the latest total used memory sample.



Note Inside a subevent description, each argument is position independent.