A Commands

This chapter describes the Cisco NX-OS system management commands that begin with the letter A.

abort (Call home)

To delete a Call home Cisco Fabric Services (CFS) configuration session without applying the configuration, use the **abort** command.

abort

Syntax Description

This command has no arguments or keywords.

Defaults

None

Command Modes

Call home configuration

SupportedUserRoles

network-admin
vdc-admin

Command History

Release	Modification
4.1(2)	This command was introduced.

Usage Guidelines

The **abort** command is supported only on the device where the CFS fabric lock is acquired.

This command does not require a license.

Examples

This example shows how to abort a Call home CFS configuration session:

```
switch(config-callhome)# abort
switch(config-callhome)# show callhome session status
Last Action Time Stamp     : Mon Dec 22 17:34:37 2008
Last Action                : Abort
Last Action Result         : Success
Last Action Failure Reason : none
```

Related Commands

Command	Description
show session status	Displays the status of the current CFS configuration session, including the last action and its result.
callhome	Enters the Call home configuration mode.
callhome distribute	Enables CFS distribution of the Call home configuration.

abort (Session Manager)

To delete a Session Manager configuration session without applying the configuration, use the **abort** command.

abort

Syntax Description

This command has no arguments or keywords.

Defaults

None

Command Modes

Session configuration

SupportedUserRoles

network-admin
vdc-admin

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to abort a Session Manager configuration session and show the aborted session:

```
switch(config-s-acl)# abort
switch# show configuration session ACL_permit_tcp
ERROR: Session not found
switch# show configuration session summary
There are no active configuration sessions
```

Related Commands

Command	Description
show configuration session	Displays the status of the current CFS configuration session, including the last action and its result.
show configuration session summary	Displays a summary of the configuration session.

action add

To specify the action of adding values of two variables when an Embedded Event Manager (EEM) applet is triggered, use the **action add** command in applet configuration mode. To undo the add action, use the **no** form of the command.

action *label* **add** {*long-integer* | *variable-name*} {*long-integer* | *variable-name*}

no action *label* **add**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
long-integer	Long integer value to be added to a variable.
variable-name	String value to be placed as variable name.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The result of **action add** is stored in the variable named $_result.

The value of the variable must be a long integer, else the action will fail.

Examples

This example shows how to add the values of two variables:

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 set $var1 10
switch(config-applet)# action 1.0 set $var2 20
switch(config-applet)# action 1.0 add $var2 $var1
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action append

To specify the action of appending a string value to a variable value when an Embedded Event Manager (EEM) applet is triggered, use the **action append** command in applet configuration mode. To undo the append action, use the **no** form of the command.

action *label* **append** *string* [*variable-value*]

no action *label* **append**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	String value to be placed as variable name.
variable-value	(Optional) Long integer value to be appended to the value of string specified.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If the variable does not exist, it will be created and set to the given value.

Examples

The following example shows how to configure an EEM applet to append given string value to the current value of the variable specified:

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 set $var1 10
switch(config-applet)# action 1.0 append $var1 20
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action break

To specify the action of exiting from a loop of actions when an Embedded Event Manager (EEM) applet is triggered, use the **action break** command in applet configuration mode. To disable the break action, use the **no** form of this command.

action *label* **break**

no action *label* **break**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use this command to skip all actions down to the related end action.

Examples

The following example shows how to configure an EEM applet to break from a loop of actions.

```
switch(config)# event manager applet loop
switch(config-applet)# event none
switch(config-applet)# action 1 while 1 eq 1
switch(config-applet)# action 2 break
switch(config-applet)# action 3 end
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action cli

To specify the action of executing a Cisco NX-OS command-line-interface (CLI) command when an Embedded Event Manager (EEM) applet is triggered, use the **action cli** command in applet configuration mode. To remove the action of executing a CLI command, use the **no** form of this command.

action *label* **cli** {**command | local command**} *cli-string* [**pattern** *pattern-string*]}

no action *label* **cli**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
command	Specifies the message to be sent to the Cisco NXOS CLI.
local	Specifies that the action has to be executed on the same card on which the event occurs.
cli-string	CLI command to be executed. If the string contains embedded blanks, enclose it in double quotation marks.
pattern *pattern-string*	(Optional) Specifies the regular expression response pattern for the **command** *cli-string* only when the command string solicits input.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

The result of the execution of this command is stored in the built-in variable $_cli_result that is set when this command is run.

The **action cli** command ends when the solicited prompt as specified in the optional **pattern** keyword is received. Specifying an incorrect pattern will cause the **action cli** command to wait forever until the applet execution times out due to the expiration of the timer.

Examples

This example shows how to specify an EEM applet to run when the **pattern** keyword specifies the *confirm* argument for the **clear counters Ethernet 0/1** command

```
switch# configure terminal
switch(config)# event manager applet cli-applet
switch(config-applet)# action 1.0 cli command enable
switch(config-applet)# action 1.2 cli command clear counters Ethernet 0/1 pattern confirm
switch(config-applet)# action 3.0 cli command y
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action comment

To specify the action of adding comments to an applet when an Embedded Event Manager (EEM) applet is triggered, use the **action comment** command in applet configuration mode. To disable the comment, use the **no** form of this command.

action *label* **comment** *string*

no action *label* **comment**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Series of characters, including embedded spaces to be placed as comment.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use this command to add comments to applets. This results in a no-operation when the applet is run.

Examples

The following example shows how to add comments to an applet.

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 cli comment keyvalue
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action continue

To specify the action of continuing with a loop of actions when an Embedded Event Manager (EEM) applet is triggered, use the **action continue** command in applet configuration mode. To stop the continue action, use the **no** form of this command.

action *label* **continue**

no action *label* **continue**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Examples

The following example shows how to configure an EEM applet to continue with a loop of actions:

```
switch(config)# event manager applet loop
switch(config-applet)# event none
switch(config-applet)# action 1 while 1 eq 1
switch(config-applet)# action 2 continue
switch(config-applet)# action 2 end
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action counter

To specify the action of setting or modifying a named counter when an Embedded Event Manager (EEM) applet is triggered, use the **action counter** command in the applet configuration mode. To restore the default value to the counter, use the **no** form of this command.

action *label* **counter name** *name* **value** *value* **op** {**dec | inc | nop | set**}

no action *label* **counter**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
name *name*	Specifies the name of the counter to be set or modified. This can be any string value. The counter name is referenced in a registered counter type policy.
value *value*	Specifies the value to be used to set or modify the counter. Integer value in the range from –2147483648 to 2147483647 inclusive.
op	Specifies the operation to be performed upon the counter.
dec	Decrements the counter by the specified *value*.
inc	Increments the counter by the specified *value*.
nop	Specifies that the counter value is read from the environment variable $_counter_value_remain.
set	Sets the counter to the value specified in *value* argument.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Use the **event counter** command with the **action counter** command when an event occurs periodically and you want an action to be implemented after a specified number of occurrences of that event.

The environment variable $_counter_value_remain is updated when the **action counter** command is completed.

Examples

This example shows how to set the counter *count1* to the value in *$variable* when the EEM *counter-applet* is triggered:

```
switch# configure terminal
switch(config)# event manager applet counter-applet
switch(config-applet)# action 1.2 counter name count1 value $variable op set
switch(config-applet)#
```

action decrement

To specify the action of decrementing the value of a variable when an Embedded Event Manager (EEM) applet is triggered, use the **action decrement** command in applet configuration mode. To remove the action from the applet, use the **no** form of the command.

action *label* **decrement** *variable-name* [*long-integer*]

no action *label* **decrement**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
variable-name	String value to be placed as variable name.
long-integer	(Optional) Long integer value by which the variable gets decremented. If it is not specified, a default value of 1 is assumed.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

None.

Examples

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 set varname 10
switch(config-applet)# action 1.0 decrement varname 3
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action divide

To divide the dividend value by the given divisor value when an Embedded Event Manager (EEM) applet is triggered, use the **action divide** command in applet configuration mode. To remove the action from the applet, use the **no** form of the command.

action *label* **divide** {*long-integer1* | *variable-name1*} {*long-integer2* | *variable-name2*}

no action *label* **divide**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
long-integer1	Dividend integer value for the division.
variable-name1	Value stored in this variable is the dividend for the division. It must be a long integer value.
long-integer2	Divisor integer value for the division.
variable-name2	Value stored in this variable is the divisor for the division. It must be a long integer value.

Defaults

None

Command Modes

Applet configuration (config-applet)

SupportedUserRoles

network-admin

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

All results of the divide action except the remainder are stored in $_result.

The remainder value of the divided integer is stored in $_remainder.

Examples

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 set varname 10
switch(config-applet)# action 1.0 divide varname 2
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action else

To identify the beginning of an else conditional action block in an if/else condition action block when an Embedded Event Manager (EEM) applet is triggered, use the **action else** command in applet configuration mode. To remove the else conditional action block, use the **no** form of the command.

action *label* **else**

no action *label* **else**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit of the configuration mode.

Examples

The following example shows how to identify the beginning of an else action block:

```
switch(config)# event manager applet one
switch(config-applet)# action 2.0 if $x eq 0
switch(config-applet)# action 3.0 else
switch(config-applet)# end
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)
action elseif	Identifies the beginning of the **elseif** conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of an **if** conditional action block when an EEM applet is triggered.

action elseif

To identify the beginning of an elseif conditional action block in an else / if conditional action block when an Embedded Event Manager (EEM) applet is triggered, use the **action elseif** command in applet configuration mode. To remove the else conditional action block, use the **no** form of the command.

action *label* **elseif** *string-op-1* {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} *string-op-2*

no action *label* **elseif**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-op-1	Sequence of characters that will replace the range of characters in the string.
eq	Compares if two strings are equal.
gt	Checks if *string-op-1* is greater than *string-op-2*
ge	Checks if *string-op-1* is greater than or equal to *string-op-2*.
lt	Checks if *string-op-1* is less than *string-op-2*.
le	Checks if *string-op-1* is less than or equal to *string-op-2*
ne	Compares if two strings are not equal.
string-op-2	Sequence of characters.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit of the configuration mode.

Examples

The following example shows how to identify the beginning of an elseif conditional action block:

```
switch(config)# event manager applet one
switch(config-applet)# event none
switch(config-applet)# action 1.0 set x "5"
switch(config-applet)# action 2.0 if $x lt 3
switch(config-applet)# action 3.0 puts $x is less than 3
switch(config-applet)# action 4.0 elseif $x lt 10
switch(config-applet)# action 5.0 puts $x is less than 10
switch(config-applet)# action 6.0 end
switch(config)# event manager run one
5 is less than 10
switch(config)#
```

Related Commands

Command	Description
action else	Identifies the beginning of the **else** conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of an **if** conditional action block when an EEM applet is triggered.
action ifgoto	Specifies the applet to jump to the given label if the condition is true when an EEM applet is triggered.

action end

To identify the end of a conditional action block in the if /else and while conditional action blocks when an Embedded Event Manager (EEM) applet is triggered, use the **action end** command in applet configuration mode. To remove the end conditional action block, use the **no** form of the command.

action *label* **end**

no action *label* **end**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Examples

The following example shows how to identify the end of a conditional action block

```
switch(config)# event manager applet one
switch(config-applet)# event none
switch(config-applet)# action 1.0 set x '5'
switch(config-applet)# action 2.0 if $x lt 10
switch(config-applet)# action 3.0 puts '$x is less than 10'
switch(config-applet)# action 4.0 end
```

Related Commands

Command	Description
action else	Identifies the beginning of the **else** conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of an **if** conditional action block when an EEM applet is triggered.

action event-default

To specify that the default action for the event is to be performed when an Embedded Event Manager (EEM) applet is triggered, use the **action event-default** command. To disable the default action, use the **no** form of this command.

action *label* **event-default**

no action *label* **event-default**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

None

Command Modes

Embedded event manager

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

If you want to allow the triggered event to process any default actions, you must configure the EEM policy to allow the default action. For example, if you match a CLI command in a match statement, you must add the **event-default** statement to the EEM policy or EEM does not allow the CLI command to execute. You can use the **terminal event-manager bypass** command to allow all EEM policies with CLI matches to execute the CLI command.

This command does not require a license.

Examples

This example shows how to specify that the default action for the event is to be performed when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet default-applet
switch(config-applet)# action 1.15 event-default
switch(config-applet)#
```

action exceptionlog

To log an exception if the specific conditions are encountered when an Embedded Event Manager (EEM) applet is triggered, use the **action exceptionlog** command. To remove the exception log, use the **no** form of this command.

action *label* **exceptionlog module** *module* **syserr** *error* **devid** *id* **errtype** *type* **errcode** *code* **phylayer** *layer* **ports** *list* **harderror** *error* [**desc** *string*]

no action *label* **exceptionlog module** *module* **syserr** *error* **devid** *id* **errtype** *type* **errcode** *code* **phylayer** *layer* **ports** *list* **harderror** *error* [**desc** *string*]

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
module *module*	Records an exception for the specified module number.
syserr *error*	Records an exception for the specified system error.
devid *id*	Records an exception for the specified device ID.
errtype *type*	Records an exception for the specified error type.
errcode *code*	Records an exception for the specified error code.
phylayer *layer*	Records an exception for the specified physical layer.
ports *list*	Records an exception for the specified ports.
harderror *error*	Records an exception for the specified hard error.
desc *string*	(Optional) Specifies a description of the exception logging condition.

Defaults

None

Command Modes

Embedded event manager

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to log an EEM applet exception:

```
switch# configure terminal
switch(config)# event manager applet exception-applet
switch(config-applet)# action 1.2 exceptionlog module 1 syserr 0x41150010 devid 96 errtype
2 errcode 354 phylayer 0 ports 1-24 harderror false desc "r2d2 general error”
switch(config-applet)#
```

action exit

To immediately exit from the running applet configuration when an Embedded Event Manager (EEM) applet is triggered, use the **action exit** command in applet configuration mode. To cancel the process of immediate exit from the running applet, use the **no** form of the command.

action *label* **exit** [*result*]

no action *label* **exit**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
result	(Optional) Parameter for the exit result.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If a statement is not associated with the applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit of the applet configuration mode.

Examples

The following example shows how to exit the applet configuration mode.

```
switch(config)# event manager applet action
switch(config-applet)# action 4.0 exit 25
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)
action else	Identifies the beginning of the **else** conditional action block when an EEEM applet is triggered.
action if	Identifies the beginning of an **if** conditional action block when an EEM applet is triggered.

action file

To configure Embedded Event Manager (EEM) applet file operations, use the **action file** command in applet configuration mode. To disable the configuration, use the **no** form of this command.

action *label* **file** {**close** *file-descriptor* | **delete** *file-descriptor* | **gets** *file-descriptor variable-name* | **open** *file-descriptor file-name access-permission* | **puts** *file-descriptor* {*string* | **nonewline** *string*} | **read** *file-descriptor variable-name* [*number*] | **write** *file-descriptor string* [*number*]}

no action *label* **file**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
close *file-descriptor*	Specifies the file to be closed.
delete *file-descriptor*	Specifies the file to be deleted.
gets *file-descriptor*	Specifies the information or the file to be fetched.
variable-name	Specifies variable to store the result of the operation.
open *file-descriptor*	Specifies the file to be opened.
file-name	Specifies the name of the file to be opened.
access-permission	Access permission of the file to be opened.
puts *file-descriptor*	Specifies the file that will be updated.
string	Data to be put in the file.
nonewline	Specifies no new line should be added.
read *file-descriptor*	Specifies the file to read.
write *file-descriptor*	Specifies the file to write to

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(1)D1(1)	This command was introduced.

Usage Guidelines

None.

Examples

The following example shows how update the file with $file as file-descriptor.

```
switch(config)# event manager applet action
switch(config-applet)# action 50 file puts $file "keywprd pair"
```

action forceshut

To configure a forced shutdown of a module, a crossbar ASIC, or the entire switch when an Embedded Event Manager (EEM) applet is triggered, use the **action forceshut** command. To remove the forced shutdown, use the **no** form of this command.

action *label* **forceshut** [{**module** *module* | **xbar** *xbar-number*}] **reset-reason** *string*

no action *label* **forceshut** [{**module** *module* | **xbar** *xbar-number*}] **reset-reason** *string*

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
module *module*	(Optional) Forces a shutdown of the specified module. The module range is from 1 to 10.
xbar *xbar-number*	(Optional) Forces a shutdown of the specified crossbar ASIC. The ASIC range is from 1 to 5.
reset-reason *string*	Provides a string that is enclosed in double quotation marks to explain the reason for a forced shutdown.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to configure a forced shutdown of module 4 when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet forceshut-applet
switch(config-applet)# action 1.3 forceshut module 4 reset-reason "module 4 failed"
switch(config-applet)#
```

action foreach

To specify the iteration of an input string using the delimiter as a tokenizing pattern, use the **action foreach** command in applet configuration mode. To remove the iteration of input string, use the **no** form of the command.

action *label* **foreach** *string-iterator string-input* [*string-delimiter*]

no action *label* **foreach**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-iterator	Series of characters that act as an iterator. If the string contains embedded blanks, enclose it in double quotes.
string-input	Series of characters that act as an input. If the string contains embedded blanks, enclose it in double quotes.
string-delimiter	(Optional) Series of characters that act as a delimiter. If the string contains embedded blanks, enclose it in double quotes. The default delimiter is whitespace.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The delimiter is a regular expression pattern string. The token found in each iteration is assigned to the given iterator variable.

If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit of the configuration.

Examples

The following example shows how to iterate an input string using the delimiter as a tokenizing pattern:

```
switch(config)# event manager applet action
switch(config-applet)# event none
switch(config-applet)# action 1.0 foreach _iterator ,red blue green orange,
switch(config-applet)# action 2.0 puts ,iterator is $_iterator,
switch(config-applet)# action 3.0 end
switch(config)# event manager run action
iterator is red iterator is blue iterator is green iterator is orange
switch(config)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action if

To identify the beginning of an if conditional action block when an Embedded Event Manager (EEM) applet is triggered, use the **action if** command in applet configuration mode. To remove the if conditional action block, use the **no** form of the command.

action *label* **if** *string-op-1* {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} *string-op-2*

no action *label* **if**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-op-1	Sequence of characters that will replace the range of characters in the string.
eq	Compares if two strings are equal.
gt	Checks if *string-op-1* is greater than *string-op-2*.
ge	Checks if *string-op-1* is greater than or equal to *string-op-2*.
lt	Checks if *string-op-1* is less than *string-op-2*.
le	Checks if *string-op-1* is less than or equal to *string-op-2*
ne	Compares if two strings are not equal.
string-op-2	Sequence of characters.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit of the configuration mode.

Examples

The following example shows how to identify the beginning of an if conditional action block:

```
switch(config)# event manager applet one
switch(config-applet)# event none
switch(config-applet)# action 1.0 set x "5"
switch(config-applet)# action 2.0 if $x lt 10
switch(config-applet)# action 3.0 puts "$x is less than 10"
switch(config-applet)# action 4.0 end
switch(config)# event manager run one
5 is less than 10
switch(config)#
```

Related Commands

Command	Description
action elseif	Identifies the beginning of the **else** conditional action block when an EEM applet is triggered.
action ifgoto	Specifies the applet to jump to the given label if the condition is true when an EEM applet is triggered.

action ifgoto

To instruct the applet to jump to a given label if the specified condition is true when an Embedded Event Manager (EEM) applet is triggered, use the **action ifgoto** command in applet configuration mode. To cancel the process of applet jump, use the **no** form of the command.

action *label-1* **if** *string-op-1* {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} *string-op-2* **goto** *label-2*

no action *label-1* **ifgoto**

Syntax Description

label-1	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-op-1	Sequence of characters that will replace the range of characters in the string.
eq	Compares if two strings are equal.
gt	Checks if *string-op-1* is greater than *string-op-2*.
ge	Checks if *string-op-1* is greater than or equal to *string-op-2*.
lt	Checks if *string-op-1* is less than *string-op-2*.
le	Checks if *string-op-1* is less than or equal to *string-op-2*.
ne	Compares if two strings are not equal.
string-op-2	Sequence of characters.
label-2	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

If the command is not specified within the applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If the **goto** *label* option is used, the **action if** command will not identify the beginning of the action block. Goto actions are supported only within the if/goto format.

To simulate a **goto** without **if,** use a test that is always true**.**

If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit of the configuration mode.

Examples

The following example shows how to instruct the applet to jump to a given label:

```
switch(config)# event manager applet one
switch(config-applet)# event none
switch(config-applet)# action 1.0 set x "5"
switch(config-applet)# action 2.0 if $x lt 10 goto 4.0
switch(config-applet)# action 3.0 puts "skipping this"
switch(config-applet)# action 4.0 puts "jumped to action 4"
switch(config-applet)# action 5.0 end
switch(config)# event manager run one
jumped to action 4
switch(config)#
```

Related Commands

Command	Description
action elseif	Identifies the beginning of the **else** conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of the **if** conditional action block when an EEM applet is triggered.

action increment

To specify the action of incrementing the value of a variable when an Embedded Event Manager (EEM) applet is triggered, use the **action increment** command in applet configuration mode. To remove the action from the applet, use the **no** form of the command.

action *label* **increment** *variable-name* [*long-integer*]

no action *label* **increment**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
increment	Increments the value of the variable with the long integer specified.
variable-name	String value placed as variable.
long-integer	(Optional) Integer value by which the variable value has to be incremented. If this value is not provided, its value is assumed to be 1.

Defaults

None

Command Modes

Applet configuration (config-applet)

SupportedUserRoles

Network-admin

Command History

Release	Modification
7.2(0)	This command was introduced.

Usage Guidelines

The value of the variable must be a long integer.

Examples

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 set var 35
switch(config-applet)# action 1.0 increment var 12
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action multiply

To specify the action of multiplying the variable value with a given integer value when an Embedded Event Manager (EEM) applet is triggered, use the **action multiply** command in applet configuration mode. To remove the action from the applet, use the **no** form of the command.

action *label* **multiply** {*long-integer1* | *variable-name1*} {*long-integer2* | *variable-name2*}

no action *label* **multiply**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
multiply	Multiplies the first parameter value by the second parameter value.
long-integer1	First integer value for the multiplication.
variable-name1	First variable for the multiplication. It must be a long integer value.
long-integer2	Second integer value for the multiplication.
variable-name2	Second variable for the multiplication. It must be a long integer value.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The result of the **action multiply** command is stored in $_result.

Examples

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 multiply 35 23
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action overbudgetshut

To configure the shutdown of a module or the entire switch due to an overbudget power condition when an Embedded Event Manager (EEM) applet is triggered, use the **action overbudgetshut** command. To remove the shutdown configuration, use the **no** form of this command.

action *label* **overbudgetshut** [**module** *module*]

no action *label* **overbudgetshut** [**module** *module*]

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
module *module*	(Optional) Forces a shutdown of the specified module. For 9slot : The range is from 1 to 9. For 10slot : The range is from 1 to 10. For 18slot : The range is from 1 to 18.

Defaults

None

(The default action is to powerdown just the linecards starting from slot_1 till the switch recovers from overbudget condition. In other words, skip both the supervisors (active and standby) and skip all spine/xbars, powerdown lcs starting from slot 1 onwards, till the "Available" power recovers from overbudget condition).

Command Modes

Applet configuration (config-applet)

SupportedUserRoles

network-admin
vdc-admin

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to configure a power overbudget shutdown of module 4 when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet overbudget-applet
switch(config-applet)# action 1.4 overbudgetshut module 4
switch(config-applet)#
```

action policy-default

To enable the default action(s) of the policy being overridden, use the **action policy-default** command. To remove the default action, use the **no** form of this command.

action *label* **policy-default**

no action *label* **policy-default**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

None

Command Modes

Applet configuration (config-applet)

SupportedUserRoles

Network-admin

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to enable the default action of a policy being overridden when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet default-applet
switch(config-applet)# action 1.65 policy-default
switch(config-applet)#
```

action publish-event

To specify the action of publishing an application-specific event when the event specified for an Embedded Event Manager (EEM) applet is triggered, use the **action publish-event** command in applet configuration mode. To disable this function, use the **no** form of the command.

action *label* **publish-event sub-system** *sub-system-id* **type** *event-type* **arg1** *argument-data* [**arg2** *argument-data*] [**arg3** *argument-data*] **[arg4** *argument-data*]

no action *label* **publish-event**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
sub-system *sub-system-id*	Specifies the identifier for the subsystem named in the *sub-system-id* argument that will publish the application event. *sub-system-id* is a number in the range of 1 to 4294967295. If the event is to be published by an EEM policy, the *sub-system-id* reserved for a customer policy is 798.
type *event-type*	Specifies the value of an event type within the specified event. Value of *event-type* is a number in the range of 1 to 4294967295.
arg1	Specifies that argument data is to be passed to the application-specific event when the event is published.
argument-data	Character text, an environment variable or a combination of the two. Optional when used with **arg2 arg3 arg4.**
arg2 arg3 arg4	(Optional) Specifies that argument data is to be passed to the application-specific event when the event is published.

Defaults

None.

Command Modes

Applet configuration (config-applet)

SupportedUserRoles

network-admin

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

None.

.

Examples

The following example shows how a policy named EventPublish_A runs every 20 seconds and publishes an event to a well-known EEM event type numbered 1. A second policy named EventPublish_B is registered to run when the well-known EEM event type 1 occurs. When policy EventPublish _B runs, it outputs a message to syslog containing the argument 1 data passed from EventPublish_A:

```
switch(config)# event manager applet EventPublish_A
switch(config-applet)# event timer watchdog time 20.0
switch(config-applet)# action 1 syslog msg "Applet EventPublish_A"
switch(config-applet)# action 2 publish-event sub-system 798 type 1 arg1 twenty
switch(config-applet)# exit
switch(config)# event manager applet EventPublish_B
switch(config-applet)# event application sub-system 798 type 1
switch(config-applet)# action 1 syslog msg "Applet EventPublish_B arg1
$_application_data1"
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action puts

To enable the action of printing data directly to the local terminal (tty) when an Embedded Event Manager (EEM) applet is triggered, use the **action puts** command in applet configuration mode. To disable this function, use the **no** form of the command.

action *label* **puts** [**nonewline**] *string*

no action *label* **puts**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
nonewline	(Optional) Suppresses the display of new line characters.
string	Sequence of characters.

Defaults

Data is not printed to the local tty.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

action puts command applies to synchronous events. The output of this command for a synchronous applet is directly displayed to the tty, bypassing the system logger (syslog).

For an asynchronous applet, the output of this command is directed to the logger.

The **nonewline** keyword suppresses the display of the newline character.

Examples

The following example shows how to print data directly to the local tty:

```
switch(config)# event manager applet puts
switch(config-applet)# event none
switch(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1
switch(config-applet)# action 2 puts "match is $_match"
switch(config-applet)# action 3 puts "submatch 1 is $_sub1"
switch(config-applet)# end
switch(config)# event manager run puts
match is one two three submatch 1 is one
switch#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)
action gets	Gets input from the local tty and stores the value in the given variable.

action reload

To specify the action of reloading the switch software when an Embedded Event Manager (EEM) applet is triggered, use the **action reload** command in the applet configuration mode. To remove the action of reloading the switch software, use the **no** form of this command.

action *label* **reload [module** *module-number***]**

no action *label* **reload**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
module *module-number*	Specifies a particular module to be reloaded.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to specify the action of reloading the switch software when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet reload-applet
switch(config-applet)# action 1.5 reload
switch(config-applet)#
```

action regexp

To match a regular expression pattern with an input string when an Embedded Event Manager (EEM) applet is triggered, use the **action regexp** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]]

no action *label* **regexp**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
regexp	Specifies the regular expression pattern that needs to be compared.
string-pattern	Sequence of characters to be used as regular expression for pattern matching.
string-input	Sequence of characters used as input string.
string-match	(Optional) Name of the variable that stores the entire pattern match.
string-submatch	(Optional) Name of the variable that stores any sub matches that are present. A maximum of three sub match strings can be specified.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The argument *string-pattern* is a regular expression. If some part of the *string-input* matches the pattern, it returns 1; otherwise it returns 0. This result is stored in the inbuilt variable $_regexp_result.

The optional *string-match* and *string-submatch* arguments store the results of the match.

Examples

This example shows how to define a regular expression match:

```
switch(config)# event manager applet regexp
switch(config-applet)# event none
switch(config-applet)# action 1.0 regexp "(.*) (.*) (.*)" "one two three" _match _sub1
switch(config-applet)# action 1.2 puts "match is $_match"
switch(config-applet)# action 1.4 puts "submatch 1 is $_sub1"
switch(config-applet)# event manager run regexp
match is one two three submatch 1 is one
switch(config)#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action set

To set the value of a variable when an Embedded Event Manager (EEM) applet is triggered, use the **action set** command in applet configuration mode. To disable this function, use the **no** form of the command.

action *label* **set** *variable-name variable-value*

no action *label* **set**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
variable-name	Name assigned to the variable to be set.
variable-value	Value of the variable.

Defaults

No variable value is set.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action set** command to set the value of a variable when an EEM applet is triggered.

Examples

The following example shows how to set the value of a variable:

```
switch(config)# event manager applet set
switch(config-applet)# event none
switch(config-applet)# action 1 set str "this is some text"
switch(config-applet)# action 2 string range "$str" 0 6
switch(config-applet)# action 3 puts "$_string_result"
switch(config-applet)# end
switch# event manager run set
this is
switch#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action snmp-trap

To specify the generation of a Simple Network Management Protocol (SNMP) trap when an Embedded Event Manager (EEM) applet is triggered, use the **action snmp-trap** command. To disable the SNMP trap, use the **no** form of this command.

action *label* **snmp-trap** [**intdata1** *integer*] [**intdata2** *integer*] [**strdata** *string*]

no action *label* **snmp-trap** [**intdata1** *integer*] [**intdata2** *integer*] [**strdata** *string*]

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
intdata1 *integer*	(Optional) Specifies an integer to be sent in the SNMP trap message to the SNMP agent.
intdata2 *integer*	(Optional) Specifies a second integer to be sent in the SNMP trap message to the SNMP agent.
strdata *string*	(Optional) Specifies a string to be sent in the SNMP trap message to the SNMP agent. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to specify an SNMP trap to generate when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet snmp-applet
switch(config-applet)# action 1.7 snmp-trap strdata "EEM detected server failure"
switch(config-applet)#
```

action string compare

To compare two unequal strings when an Embedded Event Manager (EEM) applet is triggered, use the **action string compare** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string compare [nocase] [length** *integer*] *string1 string2*

no action *label* **string compare**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
nocase	(Optional) Specifies case insensitive comparison of strings.
length *integer*	(Optional) Specifies the length of the value to limit the comparison. Value of **length** can range from 1 to 2146483647
string1	Sequence of characters to compare. If the string contains embedded blanks, enclose it in double quotation marks.
string2	Sequence of characters to compare. If the string contains embedded blanks, enclose it in double quotation marks

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

String comparisons are performed on a byte-byte basis, from left to right. The command **action string compare** forces a comparison between two unequal strings, followed by integer comparison of the result of string comparison.

When two equal strings are compared, the result is 0.When two unequal strings are compared, if the first string is longer, the result is 1 and if the second string is longer than the first, the result is -1.

The result of string comparison is stored in the inbuilt variable $_string_result.

Examples

This example shows how to compare two unequal strings:

```
switch(config)# event manager applet strcompare
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "this contains some $str"
switch(config-applet)# action 1.2 string compare nocase length 3 "contains" "$str"
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string equal

To verify if two strings are equal when an Embedded Event Manager (EEM) applet is triggered, use the **action string equal** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string equal [nocase] [length** *integer***]** *string1 string2*

no action *label* **string equal**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
nocase	(Optional) Specifies case insensitive comparison of strings.
length *integer*	(Optional) Specifies the length of the value to limit the comparison. Value of **length** can range from 1 to 2146483647.
string1	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation marks
string2	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation marks

Defaults

Strings are not verified as equal.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If two strings are equal, it returns 1.

The result of string comparison is stored in the inbuilt variable $_string_result.

Examples

This example shows how to verify if two strings are equal:

```
switch(config)# event manager applet strequal
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "this contains some data"
switch(config-applet)# action 1.2 string equal "contains" "$str"
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string first

To get the index of the first occurrence of *string1* within *string2* when an Embedded Event Manager (EEM) applet is triggered, use the **action string first** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string first** *string1 string2* [*index-value*]

no action *label* **string first**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string1	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation mark
string2	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation mark
index-value	(Optional) The index value to begin the first test. Index value is in the range of 0 to 2147483647.

Defaults

The index is not returned on the first occurrence of *string1* within *string2*.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

On the first occurrence of *string1,* the index is placed in *string2*. If *string1* is not found, it returns -1.

The result of string comparison is stored in the inbuilt variable $_string_result.

Examples

This example shows how to get the index of the first occurrence of *string1* within *string2*:

```
switch(config)# event manager applet strfirst
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "this contains some data"
switch(config-applet)# action 1.2 string first "contains" "$str"
switch(config-applet)# action 1.2 puts "$_string_result"
switch# event manager run strfirst
5
switch#
```

Related Commands

Command	Description
action string last	Returns the index of the last occurrence of *string1* within *string2*
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string index

To get the characters specified at a given index value when an Embedded Event Manager (EEM) applet is triggered, use the **action string index** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string index** *string* **[***value* | *end***]**

no action *label* **string index**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation mark
value	(Optional) The index value which is in the range of 0 to 2147483647
end	(Optional) Last character of the string.

Defaults

The characters specified at a given index value are not returned.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The index count starts from zero. Use the *end* argument for the last character of the string.

The command stores the characters in the inbuilt variable $_string_result.

Examples

This example shows how to get the character specified at a given index value:

```
switch(config)# event manager applet index
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "this is text"
switch(config-applet)# action 1.2 string index "$str" 8
switch(config-applet)# action 1.2 puts "$_string_result"
switch# event manager run index
t
switch#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string last

To get the index of the last occurrence of *string1* within *string2* when an Embedded Event Manager (EEM) applet is triggered, use the **action string last** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string last** *string1 string2* [*index-value*]

no action *label* **string last**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string1	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation mark
string2	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation mark
index-value	(Optional) The index value to begin the last test. Index value is in the range of 0 to 2147483647.

Defaults

The index is not returned on the last occurrence of *string1* within *string2*.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

On the last occurrence of *string1,* the index is placed in *string2.* If *string1* is not found, it returns -1.

The result of string comparison is stored in the inbuilt variable $_string_result.

Examples

This example shows how to get the index of the last occurrence of *string1* within *string2*:

```
switch(config)# event manager applet strlast
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "this contains some data in data file"
switch(config-applet)# action 1.2 string last "data" "$str"
switch(config-applet)# action 1.2 puts "$_string_result"
switch# event manager run strlast
28
switch#
```

Related Commands

Command	Description
action string first	Returns the index of the first occurrence of *string1* within *string2*
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string length

To get the number of characters in a string when an Embedded Event Manager (EEM) applet is triggered, use the **action string length** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string length** *string*

no action *label* **string length**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Sequence of characters to compare. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

The number of characters in a string are not returned.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The result of the command is stored the in the inbuilt variable $_string_result.

Examples

This example shows how to get the number of characters in a string:

```
switch(config)# event manager applet length
switch(config-applet)# event none
switch(config-applet)# action 1.2 string length "contains"
switch(config-applet)# action 2.0 puts "$_string_result"
switch# event manager run length
8
switch#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string match

To match a pattern of characters with a string when an Embedded Event Manager (EEM) applet is triggered, use the **action string match** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string match [nocase]** *string-pattern string*

no action *label* **string match**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
nocase	(Optional) Specifies case insensitive comparison.
string-pattern	String pattern for comparison. Use the following symbols to indicate one or more wildcards, if you use a substring: • * - to match against more than one wildcard character • ? - to match against a single wildcard character.
string	Sequence of characters to compare.If the string contains embedded blanks, enclose it in double quotation mark

Defaults

Results of pattern matching of strings are not returned to the variable $_string_result

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

If the string matches the specified pattern, result is 1; if the string does not match the specified pattern, the result is 0.

The result of the command is stored the in the inbuilt variable $_string_result.

Examples

This example shows how to match a string pattern with a string:

```
switch(config)# event manager applet match
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "NULL BBB"
switch(config-applet)# action 2.0 string match "*NULL*" "$str"
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run match
1
switch#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string range

To store a range of characters in a string when an Embedded Event Manager (EEM) applet is triggered, use the **action string range** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string range** *string start-index end-index*

no action *label* **string range**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Sequence of characters. If the string contains embedded blanks, enclose it in double quotation mark.
start-index	Starting index string value. The range is from 0 to 2147483647
end-index	Ending index string value. The range is from 0 to 2147483647.

Defaults

A string is not stored.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

The *start-index* and *end-index* arguments specify the range of characters in a string on which to operate.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to store a range of characters in a specified string:

```
switch(config)# event manager applet store
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "This is some text"
switch(config-applet)# action 2.0 string range "$string" 0 6
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run store
This is
switch#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string replace

To store a new string by replacing the range of characters in the specified string when an Embedded Event Manager (EEM) applet is triggered, use the **action string replace** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string replace** *string start-index end-index* [*new-string*]

no action *label* **string replace**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Sequence of characters. If the string contains embedded blanks, enclose it in double quotation mark.
start-index	Starting index string value. The range is from 0 to 2147483647
end-index	Ending index string value. The range is from 0 to 2147483647.
new-string	(Optional) Sequence of characters that will replace the range of characters in the string.

Defaults

A string is not stored.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action string replace** command to get a new string by replacing specific characters in a particular string when an EEM applet is triggered. If the value for new-string argument is not specified, the characters are replaced with white space.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to store a range of characters in a specified string:

```
switch(config)# event manager applet repalce
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "This is some text"
switch(config-applet)# action 2.0 string replace "$str" 0 6 "that was"
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run replace
that was some text
switch#
```

Related Commands

Command	Description
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string tolower

To store a specific range of characters of a string in lowercase when an Embedded Event Manager (EEM) applet is triggered, use the **action string tolower** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string tolower** *string* [*start-index*] [*end-index*]

no action *label* **string tolower**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Sequence of characters that needs to be changed to lower case. If the string contains embedded blanks, enclose it in double quotation mark.
start-index	(Optional) Starting index string value. The range is from 0 to 2147483647
end-index	(Optional) Ending index string value. The range is from 0 to 2147483647.

Defaults

A string is not stored.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action string tolower** command to store a specific range of characters in lower case when an EEM applet is triggered. The *start-index* and *end-index* arguments specify the range of characters on which to operate.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to convert a range of characters in a specified string, to lower case:

```
switch(config)# event manager applet lowercase
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "This is a STRING"
switch(config-applet)# action 2.0 string tolower "$str" 11 16
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run lowercase
string
switch#
```

Related Commands

Command	Description
action string toupper	Stores a specific range of characters of a string in upper case.
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string toupper

To store a specific range of characters of a string in uppercase when an Embedded Event Manager (EEM) applet is triggered, use the **action string toupper** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string toupper** *string* [*start-index*] [*end-index*]

no action *label* **string toupper**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string	Sequence of characters that needs to be changed to upper case. If the string contains embedded blanks, enclose it in double quotation mark.
start-index	(Optional) Starting index string value. The range is from 0 to 2147483647
end-index	(Optional) Ending index string value. The range is from 0 to 2147483647.

Defaults

A string is not stored.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action string toupper** command to store a specific range of characters in upper case when an EEM applet is triggered. The *start-index* and *end-index* arguments specify the range of characters on which to operate.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to convert a range of characters in a specified string, to lower case:

```
switch(config)# event manager applet uppercase
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "This is a string"
switch(config-applet)# action 2.0 string tolower "$str" 11 16
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run uppercase
STRING
switch#
```

Related Commands

Command	Description
action string tolower	Stores a specific range of characters of a string in lower case.
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string trim

To trim a string when an Embedded Event Manager (EEM) applet is triggered, use the **action string trim** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string trim** *string1* [*string2*]

no action *label* **string trim**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string1	Sequence of characters on which the trim action is to be performed. If the string contains embedded blanks, enclose it in double quotation mark.
string2	(Optional) Sequence of characters that needs to be trimmed from *string1*. If the string contains embedded blanks, enclose it in double quotation mark. Default value is whitespace.

Defaults

None.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action string trim** command to trim the characters in a string when an EEM applet is triggered. The command trims characters in *string2* from both ends of *string1*. By default, *string2* is whitespace.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to trim a string:

```
switch(config)# event manager applet trim
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "Hello How are you? Hello"
switch(config-applet)# action 2.0 string trim "$str" "Hello"
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run trim
How are you?
switch#
```

Related Commands

Command	Description
action string trimleft	Trims the characters of one string from the left end of another string.
action string trimright	Trims the characters of one string from the right end of another string.
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string trimleft

To trim the characters of one string from the left end of another string when an Embedded Event Manager (EEM) applet is triggered, use the **action string trimleft** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string trimleft** *string1* [*string2*]

no action *label* **string trimleft**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string1	Sequence of characters on which the trim action is to be performed. If the string contains embedded blanks, enclose it in double quotation mark.
string2	(Optional) Sequence of characters that needs to be trimmed from *string1*. If the string contains embedded blanks, enclose it in double quotation mark. Default value is whitespace.

Defaults

None.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action string trimleft** command to trim a string from left end of another string when an EEM applet is triggered. The command trims characters specified by*string2* from the left end of *string1*. By default, *string2* is whitespace.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to trim a string:

```
switch(config)# event manager applet trimleft
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "Hello How are you?
switch(config-applet)# action 2.0 string trimleft "$str" "Hello"
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run trimleft
How are you?
switch#
```

Related Commands

Command	Description
action string trim	Trims a string.
action string trimright	Trims the characters of one string from the right end of another string.
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action string trimright

To trim the characters of one string from the right end of another string when an Embedded Event Manager (EEM) applet is triggered, use the **action string trimright** command in applet configuration mode. To disable the function, use the **no** form of this command.

action *label* **string trimright** *string1* [*string2*]

no action *label* **string trimright**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string1	Sequence of characters on which the trim action is to be performed. If the string contains embedded blanks, enclose it in double quotation mark.
string2	(Optional) Sequence of characters that needs to be trimmed from *string1*. If the string contains embedded blanks, enclose it in double quotation mark. Default value is whitespace.

Defaults

None.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(1)	This command was introduced.

Usage Guidelines

Use the **action string trimright** command to trim a string from right end of another string when an EEM applet is triggered. The command trims characters specified by*string2* from right end of *string1*. By default, *string2* is whitespace.

The result of the command is stored the in the built-in variable $_string_result.

Examples

This example shows how to trim a string:

```
switch(config)# event manager applet trimright
switch(config-applet)# event none
switch(config-applet)# action 1.0 set str "How are you? Hello
switch(config-applet)# action 2.0 string trimright "$str" "Hello"
switch(config-applet)# action 3.0 puts "$_string_result"
switch(config-applet)# end
switch# event manager run trimright
How are you?
switch#
```

Related Commands

Command	Description
action string trim	Trims a string.
action string trimleft	Trims the characters of one string from the left end of another string.
event manager applet	Registers an applet with the Embedded Event Manager (EEM) and enters the applet configuration mode.

action subtract

To specify the action of subtracting the value of a variable from another value when an Embedded Event Manager (EEM) applet is triggered, use the **action subtract** command in applet configuration mode. To remove the action from the applet, use the **no** form of the command.

action *label* **subtract** {*long-integer1* | *variable-name1*} {*long-integer2* | *variable-name2*}

no action *label* **subtract**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
subtract	Subtracts the value of a variable from another variable.
long-integer1	First integer value for the subtraction.
variable-name1	First variable for the subtraction. It must be a long integer value.
long-integer2	Second integer value for the subtraction.
variable-name2	Second variable for the subtraction. It must be a long integer value.

Defaults

None

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use this command to subtract the value of the second parameter from the first parameter.

The result of the **action subtract** command is stored in $_result.

Examples

```
switch(config)# event manager applet one
switch(config-applet)# action 1.0 set var 37
switch(config-applet)# action 1.0 subtract var 26
switch(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)

action syslog

To configure a syslog message to generate when an Embedded Event Manager (EEM) applet is triggered, use the **action syslog** command. To disable the syslog message, use the **no** form of this command.

action *label* **syslog** [**priority** {*prio* | *prio-str*}] **msg** *msg-text*

no action *label* **syslog** [**priority** {*prio* | *prio-str*}] **msg** *msg-text*

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
priority	(Optional) Specifies the priority level of the syslog messages. If this keyword is not selected, all syslog messages are set at the informational priority level. If this keyword is selected, the priority level argument must be defined.
prio	Priority level: • **emergencies**—Specifies the system is unusable. • **alerts**—Specifies immediate action is needed. • **critical**—Specifies critical conditions. • **errors**—Specifies error conditions. • **warnings**— Specifies warning conditions. • **notifications**—Specifies normal but significant conditions. • **informational**—Specifies informational messages. This is the default. • **debugging**—Specifies debugging messages.
prio-str	A $-prefixed parameter that you previously set to a priority level.
msg *msg-text*	Specifies the message to be logged. The *msg-text* can contain character text, an environment variable, or a combination of the two. If the string contains embedded blanks, enclose it in double quotation marks.

Defaults

None

Command Modes

Embedded event manager

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

Messages written to syslog from an EEM applet are not screened for EEM syslog events, which may lead to recursive EEM syslog events. Messages that are sent from an EEM applet include the applet name for identification.

This command does not require a license.

Examples

This example shows how to configure a syslog message to save when an EEM applet is triggered:

```
switch# configure terminal
switch(config)# event manager applet syslog-applet
switch(config-applet)# action 1.7 syslog priority critical msg "Syslog condition: $log"
switch(config-applet)#
```

action while

To identify the beginning of a loop of a conditional block when an Embedded Event Manager applet is triggered, use the **action while** command in applet configuration mode. To disable this function, use the **no** form of the command.

action *label* **while** *string1 operator string2*

no action *label* **while**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in an ascending alphanumeric key sequence using the *label* as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string1	First operand.
operator	Value used with *string1* and *string2* operands to determine how the current counter value is compared to the entry value or exit value. Valid values are: • gt - greater than • ge - greater than or equal to • eq - equal to • ne - not equal to • lt - less than • le - less than or equal to
string2	Second operand.

Defaults

None.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
7.2(0)D1(1)	This command was introduced.

Usage Guidelines

Use the **action while** command to identify the beginning of a loop conditional block.

If $_variable is found within a string, it will be substituted before the expression is tested.

Examples

The following example shows how to identify the beginning of a loop conditional block when an EEM applet is triggered:

```
switch(config)# event manager applet action
switch(config-applet)# action 1.0 set _i 2
switch(config-applet)# action 2.0 while $_i lt 10
switch(config-applet)# action 3.0 action syslog msg ,i is $_i,
switch(config-applet)# action 4.0 end
```

Related Commands

Command	Description
event manager applet	Registers the applet with the Embedded Event Manager (EEM)
action else	Identifies the beginning of an else block in the if/else conditional block
action elseif	Identifies the beginning of the if/else conditional block.
action if	Identifies the beginning of an if conditional block.

alert-group

To configure a Call home CLI command for an alert group, use the **alert-group** command. To remove the command from the alert group, use the **no** form of this command.

alert-group {**All** | **Configuration** | **Diagnostic** | **EEM** |**Environmental** | **Inventory** | **License** | **Linecard-Hardware** | **Supervisor-Hardware** | **Syslog-group-port** | **System** | **Test**} **user-def-cmd** *cli_command*

no alert-group {**All** | **Configuration** | **Diagnostic** | **Environmental** | **Inventory** | **License** | **Linecard-Hardware** | **Supervisor-Hardware** | **Syslog-group-port** | **System** | **Test**} **user-def-cmd** *cli_command*

Syntax Description

All	Specifies all alert groups—configuration, diagnostic, EEM, environmental, inventory, license, linecard-hardware, supervisor-hardware, syslog-group-port, system, and test.
Configuration	Specifies events related to configurations.
Diagnostic	Specifies events related to diagnostics.
EEM	Specifies events related to EEM.
Environmental	Specifies events related to power,fan,temperature related events.
Inventory	Specifies events related to the inventory status.
License	Specifies events related to license.
Linecard-Hardware	Specifies events related to line card hardware.
Supervisor-Hardware	Specifies events related to the supervisor module.
Syslog-group-port	Specifies events related to syslog messages filed by port manager.
System	Specifies events related to software.
Test	Specifies events related to tests.
user-def-cmd *cli_command*	Configures a valid CLI command for the alert group.

Defaults

None

Command Modes

Call home configuration

SupportedUserRoles

network-admin
vdc-admin

Command History

Release	Modification
4.0(1)	This command was introduced.

Usage Guidelines

This command does not require a license.

Examples

This example shows how to add the **show ip routing** command to the configuration alert group:

```
switch# config t
switch(config)# callhome
switch(config-callhome)# alert-group Configuration user-def-cmd “show ip routing”
```

Related Commands

Command	Description
callhome	Enters the Call home configuration mode.
callhome distribute	Enables CFS distribution of the Call home configuration.
show callhome destination-profile *name*	Displays one or more Call home destination profiles.