

# repeat

Use the **repeat** ROM monitor command to repeat a command.

**repeat** [*num* | *string*]

Syntax Description	
<i>num</i>	(Optional) Number of the command (from the command history buffer list).
<i>string</i>	(Optional) String that uniquely matches a command in the command history buffer list.

**Defaults** If no argument is specified, the last command is repeated.

**Command Types** ROM monitor command.

**Command Modes** Normal.

**Usage Guidelines** The optional *num* and *string* arguments specify which command in the command history buffer to repeat. If you specify a string to match, the most recent command in the buffer to begin with the specified string is executed again.

If the string contains white space, you must use quotation marks.

This command is aliased to “r” by the ROM monitor for convenience.

**Examples** These examples show how to use the **repeat** command. You use the **history** command to display the list of previously entered commands:

```
rommon 22 > history
8  dir
9  dir bootflash:
10 dis
11 dis 0xa0001000
12 dis 0xbe000000
13 history
14 meminfo
15 meminfo -l
16 meminfo
17 meminfo -l
18 meminfo
19 meminfo
20 meminfo -l
21 meminfo -l
22 history
rommon 23 > repeat dir
dir bootflash:
      File size           Checksum   File name
1973032 bytes (0x1e1b28)  0xdadf5e24  llue
```

**repeat**

```
rommon 24 > repeat
dir bootflash:
      File size      Checksum  File name
  1973032 bytes (0x1e1b28)  0xdadf5e24  llue
rommon 25 > repeat 15
meminfo -1

Main memory size: 16 MB.
Packet memory size: 0 MB
Main memory size: 0x1000000
Available main memory starts at 0xa000e000, size 0xff2000
NVRAM size: 0x20000

Parity Map for the DRAM Banks
Socket 0 in Bank 0 Has No Parity
Socket 1 in Bank 0 Has No Parity
Socket 0 in Bank 1 Has No Parity
Socket 1 in Bank 1 Has No Parity
=====
```

## reset—ROM monitor

Use the **reset** ROM monitor command to perform a soft reset of the switch.

```
reset {mod | system}
```

Syntax Description	
<i>mod</i>	Number of the module to be reset.
<b>system</b>	Keyword that specifies to reset the entire switch.

**Defaults** This command has no default settings.

**Command Types** ROM monitor command.

**Command Modes** Normal.

**Examples** This example shows how to use the **reset** command:

```
rommon 26 > reset

System Bootstrap, Version 3.1(1.69)
Copyright (c) 1994-1997 by cisco Systems, Inc.
Supervisor III processor with 16384 Kbytes of main memory

rommon 1 >
=====
```

# reset—switch

Use the **reset** command to restart the system or an individual module and to schedule a system reset.

```
reset [mod | system | mindown]
```

```
reset [mindown] at {hh:mm} [mm/dd] [reason]
```

```
reset [mindown] in [hh:] {mm} [reason]
```

## Syntax Description

<b>mod</b>	(Optional) Number of the module to be restarted.
<b>system</b>	(Optional) Keyword to reset the system.
<b>mindown</b>	(Optional) Keyword to perform a reset as part of a minimal downtime software upgrade in a system with redundant supervisor engine modules.
<b>at</b>	Keyword to schedule a system reset at a specific future time.
<b>hh:mm</b>	Variable specifying the hour and minute of the scheduled reset.
<b>mm/dd</b>	(Optional) Month and data of scheduled reset.
<b>in</b>	Keyword to schedule a system reset in a specific time.
<b>hh:</b>	(Optional) Number of hours into the future to reset the switch.
<b>mm</b>	Number of minutes into the future to reset the switch.
<b>reason</b>	(Optional) Reason for the reset.

## Defaults

This command has no default settings.

## Command Types

Switch command.

## Command Modes

Privileged.

## Usage Guidelines

In software release 5.2 and later, you can use the **reset mindown** command to reset the switch as part of a minimal downtime software upgrade in a system with redundant supervisor engine modules. For complete information on performing a minimal downtime software upgrade, refer to the *Catalyst 5000 Family Software Configuration Guide*.



### Caution

If you make configuration changes after entering the **reset mindown** command but before the active supervisor engine resets, the changes are not saved. Input from the CLI is still accepted by the switch while the standby supervisor engine is reset, but any changes you make to the configuration between the time when you enter the **reset mindown** command and the time when the supervisor engine comes online running the new software image are not saved or synchronized with the standby supervisor engine.

If you do not specify a module number (either a switching module or the active supervisor engine), the command resets the entire system.

If you reset an intelligent module (such as the Catalyst 5000 family RSM or RSFC, or an ATM module), both the module hardware and software are completely reset.

You can use the **reset mod** command to switch to the standby supervisor engine, where *mod* is the module number of the active supervisor engine.

## Examples

This example shows how to reset the supervisor engine module on a Catalyst 5500 switch with redundant supervisor engines:

```
Console> (enable) reset 1
This command will force a switch-over to the standby supervisor module
and disconnect your telnet session.
Do you want to continue (y/n) [n]? y
Connection closed by foreign host.
host%
Console> (enable)
```

This example shows how to reset module 4:

```
Console> (enable) reset 4
This command will reset module 4 and may disconnect your telnet session.
Do you want to continue (y/n) [n]? y
Resetting module 4...
Console> (enable)
```

This example shows how to schedule a system reset for a specific future time:

```
Console> (enable) reset at 20:00
Reset scheduled at 20:00:00, Wed Aug 18 2000.
Proceed with scheduled reset? (y/n) [n]? y
Reset scheduled for 20:00:00, Wed Aug 18 2000 (in 0 day 5 hours 40 minutes).
Console> (enable)
```

This example shows how to schedule a reset for a specific future time and include a reason for the reset:

```
Console> (enable) reset at 23:00 8/18 Software upgrade to 5.3(1).
Reset scheduled at 23:00:00, Wed Aug 18 2000.
Reset reason: Software upgrade to 5.3(1).
Proceed with scheduled reset? (y/n) [n]? y
Reset scheduled for 23:00:00, Wed Aug 18 2000 (in 0 day 8 hours 39 minutes).
Console> (enable)
```

This example shows how to schedule a reset with minimum down time:

```
Console> (enable) reset mindown at 23:00 8/18 Software upgrade to 5.3(1).
Reset scheduled at 23:00:00, Wed Aug 18 2000.
Reset reason: Software upgrade to 5.3(1).
Proceed with scheduled reset? (y/n) [n]? y
Reset mindown scheduled for 23:00:00, Wed Aug 18 2000 (in 0 day 8 hours 39 minutes).
Console> (enable)
```

This example shows how to schedule a reset after a specified time:

```
Console> (enable) reset in 5:20 Configuration update
Reset scheduled in 5 hours 20 minutes.
Reset reason: Configuration update
Proceed with scheduled reset? (y/n) [n]? y
Reset scheduled for 19:56:01, Wed Aug 18 2000 (in 5 hours 20 minutes).
Reset reason: Configuration update
Console> (enable)
```

# session

Use the **session** command to access the CLI of an intelligent module such as a Catalyst 5000 family RSM, RSFC, or ATM module.

**session** *mod*

---

## Syntax Description

*mod*            Number of the ATM or RSM module.

---



---

## Defaults

This command has no default settings.

---

## Command Types

Switch command.

---

## Command Modes

Normal.

---

## Usage Guidelines

After you enter this command, the system responds with the Enter Password: prompt, if a password is configured on the module.

To end the session with the intelligent module, enter the **quit** command.

---

## Examples

This example shows how to open a session with an ATM module (module 4):

```
Console> session 4
Trying ATM-4...
Connected to ATM-4.
Escape character is '^]'.

ATM>
```

# set

Use the **set** command to display all of the ROM monitor command variable names with their values.

**set**

---

**Syntax Description** This command has no arguments or keywords.

---

**Defaults** This command has no default settings.

---

**Command Types** ROM monitor command.

---

**Command Modes** Normal.

---

**Examples** This example shows how to use the **set** command to display all of the monitor variable names with their values:

```
rommon 2 > set  
PS1=rommon ! >  
BOOT=  
?=0
```

---

**Related Commands** [varname=](#)

# set accounting commands

Use the **set accounting commands** command to enable accounting of the command events on the switch.

```
set accounting commands enable { config | enable | all } [stop-only] tacacs+
```

```
set accounting commands disable
```

Syntax Description		
<b>enable</b>	Keyword to enable the specified accounting method for commands.	
<b>config</b>	Keyword to permit accounting for configuration commands only.	
<b>enable</b>	Keyword to permit accounting for enable mode commands only.	
<b>all</b>	Keyword to permit accounting for all commands.	
<b>stop-only</b>	(Optional) Keyword to apply the accounting method at the command end.	
<b>tacacs+</b>	Keyword to specify TACACS+ accounting for commands.	
<b>disable</b>	Keyword to disable accounting for commands.	

**Defaults** Accounting is disabled by default.

**Command Types** Switch command.

**Command Modes** Privileged.

**Usage Guidelines** You must configure the TACACS+ servers before enabling accounting.

**Examples** This example shows how to enable accounting for configuration commands when sending records only upon termination of the event, using a TACACS+ server:

```
Console> (enable) set accounting commands enable config stop-only tacacs+
Accounting set to enable for commands-config events in stop-only mode.
Console> (enable)
```

This example shows how to disable command accounting:

```
Console> (enable) set accounting commands disable
Accounting set to disable for commands-config events.
Console> (enable)
```

**Related Commands**

[set accounting connect](#)  
[set accounting exec](#)  
[set accounting suppress](#)  
[set accounting system](#)  
[set accounting update](#)  
[set tacacs server](#)  
[show accounting](#)

# set accounting connect

Use the **set accounting connect** command to enable accounting of outbound connection events on the switch.

```
set accounting connect enable [start-stop | stop-only] {tacacs+ | radius}
```

```
set accounting connect disable
```

Syntax Description	enable	Keyword to enable the specified accounting method for connection events.
	<b>start-stop</b>	(Optional) Keyword to specify the accounting method applies at the start and stop of the connection event.
	<b>stop-only</b>	(Optional) Keyword to specify the accounting method applies at the conclusion of the connection event.
	<b>tacacs+</b>	Keyword to specify TACACS+ accounting for connection events.
	<b>radius</b>	Keyword to specify RADIUS accounting for connection events.
	<b>disable</b>	Keyword to disable accounting of connection events.

**Defaults** Accounting is disabled by default.

**Command Types** Switch command.

**Command Modes** Privileged.

**Usage Guidelines** You must configure the RADIUS or TACACS+ servers and shared keys before enabling accounting.

**Examples** This example shows how to enable accounting on Telnet and rlogin sessions when generating records at stop only, using a TACACS+ server:

```
Console> (enable) set accounting connect enable stop-only tacacs+
Accounting set to enable for connect events in stop-only mode.
Console> (enable)
```

This example shows how to disable accounting:

```
Console> (enable) set accounting connect disable
Accounting set to disable for connect events.
Console> (enable)
```

**Related Commands**

[set accounting commands](#)  
[set accounting exec](#)  
[set accounting suppress](#)  
[set accounting system](#)  
[set accounting update](#)  
[set tacacs server](#)  
[show accounting](#)

## set accounting exec

Use the **set accounting exec** command to enable accounting of normal mode sessions on the switch.

```
set accounting exec enable [start-stop | stop-only] { tacacs+ | radius }
```

```
set accounting exec disable
```

Syntax Description		
<b>enable</b>	Keyword to enable the specified accounting method for normal mode events.	
<b>start-stop</b>	(Optional) Keyword to specify that the specified accounting method applies at the start and stop of the normal mode event.	
<b>stop-only</b>	(Optional) Keyword to specify that the specified accounting method applies at the conclusion of the normal mode event.	
<b>tacacs+</b>	Keyword to specify TACACS+ accounting for normal mode events.	
<b>radius</b>	Keyword to specify RADIUS accounting for normal mode events.	
<b>disable</b>	Keyword to disable accounting for normal mode events.	

**Defaults** Accounting is disabled by default.

**Command Types** Switch command.

**Command Modes** Privileged.

**Usage Guidelines** You must configure the RADIUS or TACACS+ servers and shared keys before enabling accounting.

**Examples** This example shows how to enable accounting of normal login mode events when generating records at start and stop, using a RADIUS server:

```
Console> (enable) set accounting exec enable start-stop radius
Accounting set to enable for exec events in start-stop mode.
Console> (enable)
```

This example shows how to disable accounting of normal login mode events:

```
Console> (enable) set accounting exec disable
Accounting set to disable for exec events in start-stop mode.
Console> (enable)
```

**Related Commands**

[set accounting commands](#)  
[set accounting connect](#)  
[set accounting suppress](#)  
[set accounting system](#)  
[set accounting update](#)  
[set tacacs server](#)  
[show accounting](#)

# set accounting suppress

Use the **set accounting suppress** command to enable or disable suppression of accounting information for a user who has logged in without a username.

```
set accounting suppress null-username {enable | disable}
```

Syntax Description	Parameter	Description
	<b>null-username</b>	Keyword to specify unknown users.
	<b>enable</b>	Keyword to enable suppression for unknown users.
	<b>disable</b>	Keyword to disable suppression unknown users.

**Defaults** Accounting is disabled by default.

**Command Types** Switch command.

**Command Modes** Privileged.

**Usage Guidelines** You must configure the TACACS+ servers and shared keys before enabling accounting.

**Examples** This example shows how to suppress accounting information for users without a username:

```
Console> (enable) set accounting suppress null-username enable
Accounting will be suppressed for user with no username.
Console> (enable)
```

This example shows how to include accounting-event information of users without usernames:

```
Console> (enable) set accounting suppress null-username disable
Accounting will be not be suppressed for user with no username.
Console> (enable)
```

**Related Commands**

- [set accounting commands](#)
- [set accounting connect](#)
- [set accounting exec](#)
- [set accounting suppress](#)
- [set accounting system](#)
- [set accounting update](#)
- [set tacacs server](#)
- [show accounting](#)

# set accounting system

Use the **set accounting system** command to enable accounting of system events on the switch.

```
set accounting system enable [start-stop | stop-only] {tacacs+ | radius}
```

```
set accounting system disable
```

Syntax Description	
<b>enable</b>	Keyword to enable the specified accounting method for system events.
<b>start-stop</b>	(Optional) Keyword to specify the accounting method applies at the start and stop of the system event.
<b>stop-only</b>	(Optional) Keyword to specify the accounting method applies at the conclusion of the system event.
<b>tacacs+</b>	Keyword to specify TACACS+ accounting for system events.
<b>radius</b>	Keyword to specify RADIUS accounting for system events.
<b>disable</b>	Keyword to disable accounting for system events.

**Defaults** Accounting is disabled by default.

**Command Types** Switch command.

**Command Modes** Privileged.

**Usage Guidelines** You must configure the RADIUS or TACACS+ servers and shared keys before enabling accounting.

**Examples** This example shows how to enable accounting for system events when sending records only upon termination of the event, using a RADIUS server:

```
Console> (enable) set accounting system enable stop-only radius
Accounting set to enable for system events in start-stop mode.
Console> (enable)
```

This example shows how to disable accounting for system events:

```
Console> (enable) set accounting system disable
Accounting set to disable for system events.
Console> (enable)
```

**Related Commands**

[set accounting commands](#)  
[set accounting connect](#)  
[set accounting exec](#)  
[set accounting suppress](#)  
[set accounting update](#)  
[set tacacs server](#)  
[show accounting](#)