

## Chapter 24

# Scripting

Introduction .....	24-2
Creating Scripts .....	24-2
Script Commands .....	24-2
Using the Built-in Text Editor .....	24-3
Loading from a TFTP Server .....	24-3
Loading from an Asynchronous Port .....	24-3
Using Scripts .....	24-4
Script Parameters .....	24-4
Script Control Structures .....	24-5
Command Reference .....	24-5
ACTIVATE SCRIPT .....	24-6
ADD SCRIPT .....	24-7
DEACTIVATE SCRIPT .....	24-7
DELETE SCRIPT .....	24-8
IF..THEN..ELSE..ENDIF .....	24-9
SET SCRIPT .....	24-10
SHOW SCRIPT .....	24-11
WAIT .....	24-12

## Introduction

---

This chapter describes the scripting facility provided by the router, and how to create and run scripts.

The router's command processor accepts configuration commands entered from a terminal connected to an asynchronous port or a Telnet connection. The command line editing and recall functions enable previous commands to be recalled, edited and re-executed. However, this approach can be cumbersome if many similar commands have to be entered, or if sequences of commands have to be entered repeatedly at different times or on different routers.

The scripting facility enables sequences of commands to be stored in a script, and replayed at any time, allowing the router to be easily configured or quickly re-configured. Scripts can be activated either from the command line, using the `ACTIVATE SCRIPT` command on page 24-6, or from a trigger.

Scripts are stored in the router's file system as text files, either in NVS or FLASH. By convention, scripts with the file type "CFG" contain configuration commands to be executed at boot. Scripts with the file type "SCP" are intended for other repetitive tasks. A special configuration script `BOOT.CFG`, if found in either NVS or FLASH, will be executed at boot. This script allows a sequence of commands to be executed every time the router reboots.

## Creating Scripts

---

Scripts are text files containing standard router configuration commands that would normally be entered at the router's command line prompt. A script can be created using any one of the following methods:

- Using script commands entered at the router's command line prompt.
- Using the router's built-in text editor.
- Loading the file from a TFTP server using the `LOAD` command.
- Loading the file over from asynchronous port using the `LOAD` command.

## Script Commands

A script can be created from the command line, using the command:

```
ADD SCRIPT=filename [LINE=line] [TEXT=text]
```

Additional lines can be added to the script by repeating the command as often as required. The text of a line in a script file can be changed using the command:

```
SET SCRIPT=filename LINE=line TEXT=text
```

The lines in a script file can be re-ordered using either of the commands:

```
SET SCRIPT=filename LINE=line BEFORE=line  
SET SCRIPT=filename LINE=line AFTER=line
```

The contents of a script file can be displayed using the command:

```
SHOW SCRIPT [= filename]
```

The use of commands to create scripts is rather cumbersome and is not recommended, unless the script is very short or none of the other methods can be used. It is much easier to use the router's built-in text editor, or to create the file on a PC and download it using the LOAD command.

## Using the Built-in Text Editor

The router's built-in text editor can be used to create scripts. The editor is invoked using the command:

```
EDIT [filename]
```

The editor uses VT100 command sequences and should only be used with a VT100-compatible terminal, terminal emulation program or Telnet client. Scripts created using the editor must be named with a file type of "SCP" or "CFG" so they can be identified correctly by the system.



---

*Before starting the editor make sure your terminal, terminal emulation program or Telnet client is 100% compatible with a VT100 terminal.*

---

See *Chapter 1, Operation* for more information about using the built-in editor.

## Loading from a TFTP Server

Script files can be downloaded from a TFTP server using the command:

```
LOAD [FILE=filename] [DESTINATION={FLASH|NVS}] [SERVER=ipadd]  
[DELAY=delay]
```

The advantages of loading script files from a TFTP server are that the files can be created using any available plaintext editor or application that generates plain ASCII text files, scripts can be shared and used on any number of routers, and the scripts for an entire network of routers can be managed centrally under change control if required.

## Loading from an Asynchronous Port

Script files can be downloaded over one of the router's asynchronous ports using the command:

```
LOAD [FILE=filename] [DESTINATION={FLASH|NVS}] [PORT=port]  
[DELAY=delay]
```

After the LOAD command is executed, all input received via the specified asynchronous port is captured and saved in the specified file. The load stops when a control character other than a carriage return (ASCII 13) or line feed (ASCII 10) is received.

## Using Scripts

Script can in themselves activate other scripts. The newly activated child script is independent of the parent script. The two scripts will run in parallel.



*As scripts can activate other scripts, great care should be taken not to make a loop of script activation.*

To minimise the impact on the system of executing a script, a brief pause is inserted between the execution of each line of a script. The only exception to this is the boot script, BOOT.CFG, which executes commands with no delays.

The output from a script can be directed to either the TTY device (a terminal connected to an asynchronous port or a Telnet connection) from which the script was activated, or to the logging facility. The default is to direct all output from the boot script BOOT.CFG to the logging facility, and all output from other scripts to the TTY device.

## Script Parameters

Up to eight parameters can be passed to a script. Parameters are specified on the command line after the script name, separated by spaces:

```
ACTIVATE SCRIPT=[filename] [param1 param2 param3 param4
                             param5 param6 param7 param8]
```

Within a script the symbols %1 to %8 are used to refer to the passed parameters, and are automatically replaced by the parameter values before the script is executed. Parameters allow more generic scripts to be written to handle certain operations.

A number of global script parameters are available in all scripts (Table 24-1).

**Table 24-1: Global script variables.**

Variable	Meaning
%D	The current system date, in the format dd-mmm-yyyy.
%T	The current system time, in the 24 hour format hh:mm:ss.
%N	The router's system name.
%S	The router's serial number.

Parameters are passed to a script when it is executed. For instance the module-specific triggers facility passes parameters specific to each trigger type that give details of the event that activated the trigger.

---

## Script Control Structures

---

The IF . . THEN . . ELSE . . ENDIF control structure can be used execute a different set of router commands depending on some condition:

```
IF string1 {EQ|NE} string2 THEN
  router commands...
ENDIF

IF string1 {EQ|NE} string2 THEN
  router commands...
ELSE
  router commands...
ENDIF
```

The EQ and NE logical operators test that *string1* and *string2* are equal or not equal, respectively. Tests are not case sensitive, so the following expressions are equivalent:

```
FLASH EQ FLASH
FLASH EQ flash
```

If the result of the expression is true, then the router commands between the IF . . THEN and ELSE or ENDIF statements are executed. Control continues with the next statement after the IF . . THEN . . ELSE . . ENDIF statement. If the result of the expression is false and there is an ELSE clause, then the router commands between the ELSE and ENDIF statements are executed. Control continues with the next statement after the IF . . THEN . . ELSE . . ENDIF statement. If the result of the expression is false and there is no ELSE clause, then control continues with the next statement after the IF . . THEN . . ELSE . . ENDIF statement.

By using parameters in conjunction with IF . . THEN . . ELSE . . ENDIF control structures, a script can be written to behave differently depending on the values of the parameters passed to the script. For example, consider the following script, called L . SCP:

```
IF %2 EQ FLASH THEN
  LOAD FILE=%1 DEST=FLASH SERVER=202.36.163.10
ELSE
  LOAD FILE=%1 DEST=NVS SERVER=202.36.163.10
ENDIF
```

The script can be activated to load a file into FLASH with the command:

```
ACTIVATE SCRIPT=L.SCP FILE.TXT FLASH
```

---

## Command Reference

---

This section describes the commands available on the router to configure the script facility, and to create and execute scripts.

See “Conventions” on page lxxi of *Preface* at the front of this manual for details of the conventions used to describe command syntax. See *Appendix B, Messages* for a complete list of all messages and their meanings.

# ACTIVATE SCRIPT

---

**Syntax** ACTIVATE SCRIPT=*filename* [OUTPUT=*device*] [*parameters*]

where:

- *filename* is a file name of the form `device:filename.type`. "device" is the name of the memory device in which the file is stored (e.g. FLASH, NVS). "type" must be "SCP" or "CFG". If "type" is not entered, "SCP" will be assumed. Valid characters are uppercase letters (A–Z), lowercase letters (a–z), digits (0–9) and the hyphen character (-). Wildcards are not allowed.
- *device* is the name of the device to which the output from the script will be directed (e.g. LOG).
- *parameters* is a list of one to eight parameters. Each parameter is a character string, 1 to 255 characters in length. Valid characters are any printable character.

**Description** This command activates the playing of a script file. The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename and type. The type must be "SCP" or "CFG".

The OUTPUT parameter specifies the name of the device to which the output from the script will be directed. The only output device currently supported is the logging facility (LOG). If OUTPUT is not specified, the output from the script is sent to the TTY device (a terminal connected to an asynchronous port or a Telnet connection) from which the script was activated.

Up to eight parameters can be passed to a script. Parameters are specified on the command line after the script name, separated by spaces. Within the script, the parameters are referenced by the symbols %1 to %8, which are replaced at run time by the parameter values.



---

*For security reasons this command will only be accepted if the user has SECURITY OFFICER privilege.*

---

**Examples** To activate a script called SHOWME.SCP stored in NVS, use the command:

```
ACTIVATE SCRIPT=NVS:SHOWME.SCP
```

**See Also** ADD SCRIPT  
DELETE SCRIPT  
DEACTIVATE SCRIPT  
SET SCRIPT  
SHOW SCRIPT

---

## ADD SCRIPT

---

**Syntax** `ADD SCRIPT=filename TEXT=text [LINE=line]`

where:

- *filename* is a file name of the form `device:filename.type`. "device" is the name of the memory device in which the file is stored (e.g. FLASH, NVS). "type" must be "SCP" or "CFG". If "type" is not entered, "SCP" will be assumed. Valid characters are uppercase letters (A–Z), lowercase letters (a–z), digits (0–9) and the hyphen character (-). Wildcards are not allowed.
- *text* is a character string, 1 to 127 characters in length. Valid characters are any printable character. If the string contains spaces it must be enclosed in double quotes.
- *line* is the number of a line in the script, expressed as a decimal number.

**Description** This command adds a line of text to an existing script. The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename and type. The type must be "SCP" or "CFG".

The TEXT parameter specifies the line of text to add to the script.

The LINE parameter specifies the line in the script after which the new line of text will be inserted. If the LINE parameter is not specified, the new line of text will be added to the end of the script.



---

*For security reasons this command will only be accepted if the user has SECURITY OFFICER privilege.*

---

**Examples** To add a script called SHOWME.SCP stored in NVS, use the command:

```
ADD SCRIPT=NVS:SHOWME.SCP TEXT="SHOW LOG"
```

**See Also** `ACTIVATE SCRIPT`  
`DELETE SCRIPT`  
`DEACTIVATE SCRIPT`  
`SET SCRIPT`  
`SHOW SCRIPT`  
`WAIT`

---

## DEACTIVATE SCRIPT

---

**Syntax** `DEACTIVATE SCRIPT=filename`

where:

- *filename* is a file name of the form `device:filename.type`. "device" is the name of the memory device in which the file is stored (e.g. FLASH, NVS). "type" must be "SCP" or "CFG". If "type" is not entered, "SCP" will be assumed. Valid characters are uppercase letters (A–Z), lowercase letters (a–z), digits (0–9) and the hyphen character (-). Wildcards are not allowed.

**Description** This command stops the playing of a script file.

The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename and type. The type must be "SCP" or "CFG". Because of the speed that scripts play, and their generally small size, it may not be practical to stop a script once it has been activated.




---

*For security reasons this command will only be accepted if the user has SECURITY OFFICER privilege.*

---

**Examples** To deactivate a script called SHOWME.SCP that was stored in NVS, use the command:

```
DEACTIVATE SCRIPT=NVS:SHOWME.SCP
```

**See Also** ACTIVATE SCRIPT  
ADD SCRIPT  
DELETE SCRIPT  
SET SCRIPT  
SHOW SCRIPT

## DELETE SCRIPT

---

**Syntax** DELETE SCRIPT=*filename* [LINE=*line*]

where:

- *filename* is a file name of the form `device:filename.type`. "device" is the name of the memory device in which the file is stored (e.g. FLASH, NVS). "type" must be "SCP" or "CFG". If "type" is not entered, "SCP" will be assumed. Valid characters are uppercase letters (A–Z), lowercase letters (a–z), digits (0–9) and the hyphen character (-). Wildcards are not allowed.
- *line* is the number of a line in the script, expressed as a decimal number.

**Description** This command deletes an entire script file, or deletes a line of text from a script file.

The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename and type. The type must be "SCP" or "CFG".

The LINE parameter specifies the line in the script to be deleted. If the LINE parameter is not specified, the entire script is deleted.




---

*For security reasons this command will only be accepted if the user has SECURITY OFFICER privilege.*

---

**Examples** To delete a script called SHOWME.SCP stored in NVS, use the command:

```
DELETE SCRIPT=NVS:SHOWME.SCP
```

**See Also** ACTIVATE SCRIPT  
ADD SCRIPT  
DEACTIVATE SCRIPT  
DELETE FILE

```
SET SCRIPT
SHOW SCRIPT
```

## IF..THEN..ELSE..ENDIF

---

**Syntax** IF *string1* {EQ|NE} *string2* THEN *commands* [ELSE *commands*]  
ENDIF

where:

- *string1* and *string2* are character strings, 1 to 255 characters in length. Valid characters are any printable character.

**Description** The IF..THEN..ELSE..ENDIF control structure is used in a script to execute a different set of router commands depending on some condition:

```
IF string1 {EQ|NE} string2 THEN
    router commands...
ENDIF

IF string1 {EQ|NE} string2 THEN
    router commands...
ELSE
    router commands...
ENDIF
```

The EQ and NE logical operators test that *string1* and *string2* are equal or not equal, respectively. Tests are not case sensitive, so the expressions:

```
FLASH EQ FLASH
FLASH EQ flash
```

are equivalent. *string1* and *string2* may be the replaceable parameters %1 to %8, allowing script execution to be controlled by parameters passed to the script.

If the result of the expression is true, then the router commands between the IF..THEN and ELSE or ENDIF statements are executed. Control continues with the next statement after the IF..THEN..ELSE..ENDIF statement. If the result of the expression is false and there is an ELSE clause, then the router commands between the ELSE and ENDIF statements are executed. Control continues with the next statement after the IF..THEN..ELSE..ENDIF statement. If the result of the expression is false and there is no ELSE clause, then control continues with the next statement after the IF..THEN..ELSE..ENDIF statement.

**Examples** The following script, named L.SCP, illustrates conditional execution based on passed parameters:

```
IF %2 EQ FLASH THEN
    LOAD FILE=%1 DEST=FLASH SERVER=202.36.163.10
ELSE
    LOAD FILE=%1 DEST=FLASH SERVER=202.36.163.10
ENDIF
```

The script could be activated to load the file FILE.TXT into FLASH using the command:

```
ACTIVATE SCRIPT=L.SCP FILE.TXT FLASH
```

**See Also** WAIT

# SET SCRIPT

---

**Syntax** SET SCRIPT=*filename* LINE=*line* [AFTER=*line*] [BEFORE=*line*]  
[TEXT=*text*]

where:

- *filename* is a file name of the form `device:filename.type`. "device" is the name of the memory device in which the file is stored (e.g. FLASH, NVS). "type" must be "SCP" or "CFG". If "type" is not entered, "SCP" will be assumed. Valid characters are uppercase letters (A–Z), lowercase letters (a–z), digits (0–9) and the hyphen character (-). Wildcards are not allowed.
- *line* is the number of a line in the script, expressed as a decimal number.
- *text* is a character string, 1 to 127 characters in length. Valid characters are any printable character. If the string contains spaces it must be enclosed in double quotes.

**Description** This command is used to change the contents of a script file, in command line mode.

The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename and type. The type must be "SCP" or "CFG".

The LINE parameter specifies the line in the script to be replaced or moved. If the LINE parameter is used with the TEXT parameter, the LINE parameter specifies the line to be replaced and the TEXT parameter specifies the new contents of the line. If the LINE parameter is used with the AFTER or BEFORE parameters, the LINE parameter specifies the line to be moved and the AFTER or BEFORE parameter specifies the new position of the line in the script. One and only one of the parameters AFTER, BEFORE or TEXT must be specified, in addition to the LINE parameter. The parameters AFTER, BEFORE and TEXT are mutually exclusive. A line can be moved or changed, but not moved and changed, in a single command.

The AFTER parameter specifies the new location of the line identified by the LINE parameter in the script file. The line specified by the LINE parameter is moved to the line following the line specified by the AFTER parameter.

The BEFORE parameter specifies the new location of the line identified by the LINE parameter in the script file. The line specified by the LINE parameter is moved to the line immediately preceding the line specified by the BEFORE parameter.

The TEXT parameter specifies the new contents of the line identified by the LINE parameter in the script file. The entire line is replaced.

There are easier methods of changing scripts than using the SET SCRIPT command. Script files can be edited using the built-in editor (see *Chapter 1, Operation*), or edited on another computer system and downloaded to the router using the LOAD command.



---

*For security reasons this command will only be accepted if the user has SECURITY OFFICER privilege.*

---

**Examples** To change the third line of text in a script called SHOWME.SCP stored in NVS, use the command:

```
SET SCRIPT=NVS:SHOWME.SCP LINE=3 TEXT="SHOW TIME"
```

**See Also** ACTIVATE SCRIPT  
ADD SCRIPT  
DELETE SCRIPT  
DEACTIVATE SCRIPT  
SHOW SCRIPT

## SHOW SCRIPT

**Syntax** SHOW SCRIPT [=filename]

where:

- *filename* is a file name of the form `device:filename.type`. "device" is the name of the memory device in which the file is stored (e.g. FLASH, NVS). "type" must be "SCP" or "CFG". If "type" is not entered, "SCP" will be assumed. Valid characters are uppercase letters (A–Z), lowercase letters (a–z), digits (0–9) and the hyphen character (-). Wildcards are not allowed.

**Description** This command displays the list of scripts stored on the router, or the contents of the specified script file.

The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename and type. The type must be "SCP" or "CFG". If a file name is not specified then the list of all scripts stored on the router is displayed (Figure 24-1 on page 24-11, Table 24-2 on page 24-12). If a file name is specified then the contents of that script file are displayed (Figure 24-2 on page 24-12).

**Figure 24-1: Example output from the SHOW SCRIPT command.**

Configuration Scripts:			
Filename	Device	Size	Created
-----	-----	-----	-----
boot.cfg	nvs	127	18-May-1996 11:08:10
-----	-----	-----	-----
General Scripts:			
Filename	Device	Size	Created
-----	-----	-----	-----
acctstd.scp	nvs	201	30-May-1996 15:15:39
syn-trig.scp	nvs	1910	30-May-1996 14:41:16
syn0-a1.scp	nvs	456	22-May-1996 14:11:03
test.scp	nvs	13	05-Jun-1996 12:18:56
-----	-----	-----	-----

**Figure 24-2: Example output from the SHOW SCRIPT command for a specified script.**

```

File : nvs:acctstd.scp

1:purge acc
2:set po=2 sp=115.2k cd=connect flow=hard
3:add acc call=dialin dir=ans auth=none encap=none po=2
4:add acc script=nvs:reset.mds text="[ate0q1s0=1^M]"
5:set acc call=dialin rscr=nvs:reset.mds

```

**Table 24-2: Parameters displayed in the output of the SHOW SCRIPT command.**

Parameter	Meaning
Filename	The file name of the script file.
Device	The memory device on the router in which the script file is stored; one of "nvs" or "flash".
Size	The size of the script file, in bytes.
Created	The date and time the script file was created.

**Examples** To display the list of scripts stored on the router, use the command:

```
SHOW SCRIPT
```

**See Also** ACTIVATE SCRIPT  
ADD SCRIPT  
DEACTIVATE SCRIPT  
DELETE SCRIPT  
SET SCRIPT

## WAIT

**Syntax** WAIT *delay*

where:

- *delay* is a time delay, in seconds.

**Description** This command pauses execution of the active script for the specified period of time. The WAIT command is only valid when executed from a script, and can not be executed directly from the command line.

**Examples** To pause the active script for five seconds, use the command:

```
WAIT 5
```

**See Also** IF..THEN..ELSE..ENDIF