Chapter 34

Scripting

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 when many similar commands must be entered, or when sequences of commands must be repeatedly entered 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 from the command line (using the **activate script** command on page 34-6), from a trigger, or (for autoexec.scp) when a user with user privilege logs in.

Scripts are stored in the router's file system as text files in flash. By convention, scripts with a CFG file type contain configuration commands to be executed at boot. Scripts with a SCP file type are intended for other repetitive tasks. A special configuration script BOOT.CFG if in flash is executed at boot. This script executes a sequence of commands every time the router reboots. An AUTOEXEC.SCP script can also be created and loaded to flash. This script runs when a user with User privileges logs in. For example, this script can automatically establish a Telnet or Telbin session to a remote host when the user logs on instead of showing the command prompt.

Creating Scripts

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

- **Using Script Commands**
- **Using the Built-In Text Editor**
- **Loading from a TFTP Server**
- **Loading from an Asynchronous Port**

When a script includes more than one command, the commands are executed in sequence and as quickly as possible, without necessarily waiting for the previous command to finish. Therefore, when running scripts with multiple commands, the router may interpret the output of one command as input for a subsequent command. When creating new scripts, users should test them to make sure they run as desired.

Using Script Commands

Script commands are entered at the router's command line prompt. To create a script, use the **add script** command on page 34-7.

Additional lines can be added to the script by repeating the command as often as necessary. To change text in a script file, use the **set script** command on page 34-11.

To re-order lines in a script file, use one of these commands:

```
set script=filename line=line before=line
set script=filename line=line after=line
```

To display the contents of a script file, use the **show script** command on page 34-12.

Using commands to create scripts can be cumbersome and is recommended only when the script is short or no other method is available. 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 by using the command:

```
edit [filename]
```

See "The Built-in Editor" on page 1-44 of Chapter 1, Operation for more information about using the built-in editor. Before starting the editor, make sure your terminal, terminal emulation program, or Telnet client is 100% compatible with a VT100 terminal.

The editor uses VT100 command sequences and should be used with a VT100-compatible terminal, terminal emulation program, or Telnet client. Scripts created with the editor must be named as a SCP or CFG file type so the system can identify them.

Loading from a TFTP Server

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

```
load [file=filename] [destination={bootblock|flash}]
   [server=ipadd] [delay=delay]
```

The advantages of loading script files from a TFTP server are that the files can be created with any text editor or application that generates plain ASCII text files. Also, 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 desired.

Loading from an Asynchronous Port

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

```
load [file=filename] [destination={bootblock|flash}]
    [asyn=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

Scripts can activate other scripts. A newly activated child script is independent of the parent and runs in parallel.

Because scripts can activate other scripts, take great care 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 exception is the boot script, BOOT.CFG, which executes commands with no delays.

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

When using large scripts and if the command line option "Q=Quit" is entered, a wait may be required while the script is executed. The words "Script activating. Please wait..." appears. When the script finishes executing, "Script complete" appears along with the command prompt.

Script Parameters

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

Up to eight parameters can be passed to a script in the **activate script** command on page 34-6. Parameters should be separated by spaces, for example:

```
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 34-1).

Table 34-1: 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 system name for the router.
%S	The serial number for the router.

Script Control Structures

The `IF..THEN..ELSE..ENDIF` control structure can be used 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. Note that the test, or comparison, of *string1* and *string2* is limited to the first three characters of each string. For example, consider the following script:

```
if %N eq tsta then
   set sys name=tstb

else
   set sys name=tsta
endif
```

Using this script would not change the system name as the characters *tst* in each string are compared, not *tsta* and *tstb*. 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 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`:

```
load file=%1 DEST=flash ser=202.36.163.10
if %2 eq go then
    script=%1 %3 %4 %5 %6 %7 %8
endif
```

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

```
activate script=l.scp linkup.scp go ppp0
```

Command Reference

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

The shortest valid command is denoted by capital letters in the Syntax section. See "Conventions" on page xcv of Preface at the front of this manual for details of the conventions used to describe command syntax. See Appendix A, 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 in the format `[device:]filename.type`.
    - *device* is the name of the memory device where the file is stored – flash. If *device* is specified, it must be separated from the rest of the file name by a colon (:). The default is flash.
    - *type* must be .SCP or .CFG. If *type* is not entered, .SCP is assumed.
    - Valid characters are uppercase and lowercase letters, digits (0–9), and the characters ~ ' ! @ # $ % ^ & ( ) _ - { }. Invalid characters are * + = " | \ [ ] ; : ? / , < >. Wildcards are not allowed.
- *device* is the name of the device where the output from the script is to be directed (for example, LOG).
- *parameters* is a list of one to eight parameters. Each parameter is a character string 1 to 255 characters long. Valid characters are any printable character.

Description This command activates the playing of a script file, and can be issued only by a user with Security Officer privilege.

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 where output from the script is directed. The only output device currently supported is the Logging facility (LOG). If OUTPUT is not specified, output goes to the TTY device (a terminal connected to an asynchronous port or a Telnet connection) where 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.

Examples To activate a script called `SHOWME.SCP`, use the command:

```
act sc=showme.scp
```

Related Commands

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 in the format `[device:]filename.type`.
    - *device* is the name of the memory device where the file is stored – flash. If *device* is specified, it must be separated from the rest of the file name by a colon (:). The default is flash.
    - *type* must be .SCP or .CFG. If *type* is not entered, .SCP is assumed.
    - Valid characters are uppercase and lowercase letters, digits (0–9), and the characters ~ ’ ! @ # $ % ^ & ( ) _ - { }. Invalid characters are * + = “ | \ [ ] ; : ? / , < >. Wildcards are not allowed.
- *text* is a character string 1 to 127 characters long. Valid characters are any printable character. If the string contains spaces, it must be 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, and can be issued only by a user with Security Officer privilege.

The SCRIPT parameter specifies the file name of the script. A complete file name must be specified, including device, filename, and type. The file 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 is inserted. If the LINE parameter is not specified, the new line of text is added to the end of the script.

Examples To add a script called `SHOWME.SCP`, use the command:

```
add sc=showme.scp tex="show log"
```

Related Commands

activate script
delete script
deactivate script
set script
show script
wait

deactivate script

Syntax

```
DEACTivate SCript=filename
```

where:

- *filename* is a file name in the format `[device:]filename.type`.
    - *device* is the name of the memory device where the file is stored – flash. If *device* is specified, it must be separated from the rest of the file name by a colon (:). The default is flash.
    - *type* must be .SCP or .CFG. If *type* is not entered, .SCP is assumed.
    - Valid characters are uppercase and lowercase letters, digits (0–9), and the characters ~ ' ! @ # $% ^ & ( ) _ - { }. Invalid characters are * + = " | \ [ ] ; : ? / , < >. Wildcards are not allowed.

Description This command stops the playing of a script file, and can be issued only by a user with Security Officer privilege.

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

Examples To deactivate a script called `SHOWME.SCP`, use the command:

```
deact sc=showme.scp
```

Related Commands

activate script
add script
delete script
set script
show script

delete script

Syntax

```
DELete SCript=filename [LIne=line]
```

where:

- *filename* is a file name in the format `[device:]filename.type`.
    - *device* is the name of the memory device where the file is stored – flash. If *device* is specified, it must be separated from the rest of the file name by a colon (:). The default is flash.
    - *type* must be .SCP or .CFG. If *type* is not entered, .SCP is assumed.
    - Valid characters are uppercase and lowercase letters, digits (0–9), and the characters ~ ' ! @ # $% ^ & ( ) _ - { }. Invalid characters are * + = " | \ [ ] ; : ? / , < >. 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. It can be issued only by a user with Security Officer privilege.

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

Examples

To delete a script called `SHOWME.SCP`, use the command:

```
del sc=showme.scp
```

Related Commands

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 long. 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:

```
act sc=l.scp file.txt fl
```

Related Commands

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 where the file is stored – flash. If *device* is specified, it must be separated from the rest of the file name by a colon (:). The default is flash.
    - *type* must be .SCP or .CFG. If *type* is not entered, .SCP is assumed.
    - Valid characters are uppercase and lowercase letters, digits (0–9), and the characters ~ ' ! @ # $% ^ & ( ) _ - { }. Invalid characters are * + = " | \ [ ] ; : ? / , < >. 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 long. Valid characters are any printable character. If the string contains spaces, it must be in double quotes.

Description

This command is used to change the contents of a script file while in command line mode. It can be issued only by a user with Security Officer privilege.

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

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

```
set sc=showme.scp li=3 te="show time"
```

Related Commands

activate script
add script
delete script
deactivate script
show script

show script

Syntax

```
SHow SCript[=filename]
```

where:

- *filename* is a file name in the format `[device:]filename.type`.
    - *device* is the name of the memory device where the file is stored – flash. If *device* is specified, it must be separated from the rest of the file name by a colon (:). The default is flash.
    - *type* must be .SCP or .CFG. If *type* is not entered, .SCP is assumed.
    - Valid characters are uppercase and lowercase letters, digits (0–9), and the characters ~ ' ! @ # $% ^ & ( ) _ - { }. Invalid characters are * + = " | \ [ ] ; : ? / , < >. 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 file 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 (see Figure 34-1 on page 34-13 and Table 34-2 on page 34-13). If a file name is specified then the contents of that script file are displayed (see Figure 34-2 on page 34-13 and Table 34-2 on page 34-13).

Figure 34-1: Example output from the **show script** command

```
Configuration Scripts:

Filename                Device        Size     Created                Locks
-----------------------------------------------------------------------------
boot.cfg                flash         127      18-May-1996 11:08:10     0
sixteenalongfile.cfg    flash         124      30-May-1996 15:10:12     0
-----------------------------------------------------------------------------

General Scripts:

Filename                Device        Size     Created                Locks
-----------------------------------------------------------------------------
acctstd.scp             flash         201      30-May-1996 15:15:39     0
syn-trig.scp            flash         1910     30-May-1996 14:41:16     0
syn0-a1.scp             flash         456      22-May-1996 14:11:03     0
test.scp                flash         13       05-Jun-1996 12:18:56     0
sixteenalongfile.scp    flash         24       30-May-1996 15:10:12     0
-----------------------------------------------------------------------------
```

Figure 34-2: Example output from the **show script** command for a specific script

```
File : flash:create ppp.scp

1:create ppp=0 over=syn0
2:enable ip
3:add ip int=eth0  ip=192.168.10.1
4:add ip int=ppp0  ip=192.168.254.1
5:add ip route=0.0.0.0 next=0.0.0.0 int=ppp0
```

Table 34-2: Parameters in the output of the **show script** command

Parameter	Meaning
Filename	The file name of the script file.
Device	Whether the memory device on the router where the script file is stored is flash.
Size	The byte size of the script file.
Created	The date and time the script file was created.
Locks	The number of concurrent processes using the file.

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

```
sh sc
```

Related Commands

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 valid when executed from a script and cannot be executed directly from the command line.

Examples

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

```
wait 5
```

Related Commands

if..then..else..endif