Creating and Using Modem Chat Scripts

This chapter describes how to create and use modem chat scripts. These tasks are presented in the following main sections:

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Supported Platforms" section in the "Using Cisco IOS Software" chapter.

For a complete description of the modem support commands in this chapter, refer to the *Cisco IOS Dial Technologies Command Reference* publication. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

Chat Script Overview

Chat scripts are strings of text used to send commands for modem dialing, logging in to remote systems, and initializing asynchronous devices connected to an asynchronous line.

Note On a router, chat scripts can be configured only on the auxiliary port.

A chat script must be configured to dial out on asynchronous lines. You also can configure chat scripts so that they can be executed automatically for other specific events on a line, or so that they are executed manually.

Each chat script is defined for a different event. These events can include the following:

- Line activation
- Incoming connection initiation
- Asynchronous dial-on-demand routing (DDR)
- Line resets

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

- Startup

Note Outbound chat scripts are not supported on lines where modem control is set for inbound activity only using the **modem dialin** command.

How To Configure Chat Scripts

The following tasks must be performed before a chat script can be used:

- Define the chat script in global configuration mode using the **chat-script** command.
- Configure the line so that a chat script is activated when a specific event occurs (using the **script** line configuration command), or start a chat script manually (using the **start-chat** privileged EXEC command).

To configure a chat script, perform the tasks in the following sections:

- Understanding Chat Script Naming Conventions (Required)
- Creating a Chat Script (Required)
- Configuring the Line to Activate Chat Scripts (Required)
- Manually Testing a Chat Script on an Asynchronous Line (Optional)

See the section "Using Chat Scripts" later in this chapter for examples of how to use chat scripts.

Understanding Chat Script Naming Conventions

When you create a script name, include the modem vendor, type, and modulation, separated by hyphens, as follows:

vendor-type-modulation

For example, if you have a Telebit t3000 modem that uses V.32*bis* modulation, your script name would be*:*

```
telebit-t3000-v32bis
```

Note Adhering to the recommended naming convention allows you to specify a range of chat scripts by using partial names in UNIX-style regular expressions. The regular expressions are used to match patterns and select chat scripts to use. This method is particularly useful for dialer rotary groups on an interface that dials multiple destinations. Regular expressions are described in the "Regular Expressions" appendix in the *Cisco IOS Terminal Services Configuration Guide*.

Creating a Chat Script

We recommend that one chat script (a "modem" chat script) be written for placing a call and that another chat script (a "system" or "login" chat script) be written to log in to remote systems, where required.

To define a chat script, use the following command in global configuration mode:

Command	Purpose
`Router(config)# chat-script script-name expect send...`	Creates a script that will place a call on a modem, log in to a remote system, or initialize an asynchronous device on a line.

The Cisco IOS software waits for the string from the modem (defined by the *expect* portion of the script) and uses it to determine what to send back to the modem (defined by the *send* portion of the script).

Chat String Escape Key Sequences

Chat script send strings can include the special escape sequences listed in Table 1.

Table 1 Chat Script Send String Escape Sequences

Escape Sequence	Description
\	Sends the ASCII character with its octal value.
\\	Sends a backslash (\) character.
\”	Sends a double-quote (“) character (does not work *within* double quotes).
\c	Suppresses a new line at the end of the send string.
\d	Delays for 2 seconds.
\K	Inserts a BREAK.
\n	Sends a newline or linefeed character.
\N	Sends a null character.
\p	Pauses for 0.25 second.
\q	Reserved, not yet used.
\r	Sends a return.
\s	Sends a space character.
\t	Sends a tab character.
\T	Replaced by phone number.
“ ”	Expects a null string.
BREAK	Causes a BREAK. This sequence is sometimes simulated with line speed changes and null characters. May not work on all systems.
EOT	Sends an end-of-transmission character.

Adding a Return Key Sequence

After the connection is established and you press the Return key, you must often press Return a second time before the prompt appears. To create a chat script that enters this additional Return key for you, include the following string with the Return key escape sequence (see Table 1) as part of your chat script:

```
ssword:-/r-ssword
```

This part of the script specifies that, after the connection is established, you want **ssword** to be displayed. If it is not displayed, you must press Return again after the timeout passes. (For more information about expressing characters in chat scripts, see the "Regular Expressions" appendix in the *Cisco IOS Terminal Services Configuration Guide*.)

Chat String Special-Case Script Modifiers

Special-case script modifiers are also supported; refer to Table 2 for examples.

Table 2 Special-Case Script Modifiers

Special Case	Function
ABORT *string*	Designates a string whose presence in the input indicates that the chat script has failed. (You can have as many active abort entries as you like.)
TIMEOUT *time*	Sets the time to wait for input, in seconds. The default is 5 seconds, and a timeout of 60 seconds is recommended for V.90 modems.

For example, if a modem reports BUSY when the number dialed is busy, you can indicate that you want the attempt stopped at this point by including ABORT BUSY in your chat script.

Note If you use the *expect-send* pair ABORT SINK instead of ABORT ERROR, the system terminates abnormally when it encounters SINK instead of ERROR.

Configuring the Line to Activate Chat Scripts

Chat scripts can be activated by any of five events, each corresponding to a different version of the **script** line configuration command. To start a chat script manually at any point, see the following section, "Manually Testing a Chat Script on an Asynchronous Line."

To define a chat script to start automatically when a specific event occurs, use one of the following commands in line configuration mode:

Command	Purpose
`Router(config-line)# script activation` *regexp*[1]	Starts a chat script on a line when the line is activated (every time a command EXEC is started on the line).
`Router(config-line)# script connection` *regexp*	Starts a chat script on a line when a network connection is made to the line.
`Router(config-line)# script dialer` *regexp*	Specifies a modem script for DDR on a line.
`Router(config-line)# script reset` *regexp*[2]	Starts a chat script on a line whenever the line is reset.
`Router(config-line)# script startup` *regexp*[2]	Starts a chat script on a line whenever the system is started up.

1. The *regexp* argument is a regular expression that is matched to a script name that has already been defined using the **chat-script** command.
2. Do not use the **script reset** or **script startup** commands to configure a modem; instead use the **modem autoconfigure** command.

Note Outbound chat scripts are not supported on lines where modem control is set for inbound activity only (using the **modem dialin** command).

Manually Testing a Chat Script on an Asynchronous Line

To test a chat script on any line that is currently not active, use the following commands in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# **debug chat line** *number*`	Starts detailed debugging on the specified line.
Step 2	`Router# **start-chat** *regexp* [*line-number* [*dialer-string*]]`	Starts a chat script on any asynchronous line.

If you do not specify the line number, the script runs on the current line. If the line specified is already in use, you cannot start the chat script. A message appears indicating that the line is already in use.

Using Chat Scripts

The following sections provide examples of how to use chat scripts:

- Generic Chat Script Example
- Traffic-Handling Chat Script Example
- Modem-Specific Chat Script Examples
- Dialer Mapping Example
- System Login Scripts and Modem Script Examples

Generic Chat Script Example

The following example chat script includes a pair of empty quotation marks (“ ”), which means “expect anything,” and \r, which means “send a return”:

```
" " \r "name:" "myname" "ord":" "mypassword" ">" "slip default"
```

Traffic-Handling Chat Script Example

The following example shows a configuration in which, when there is traffic, a random line will be used. The dialer code will try to find a script that matches either the modem script *.*-v32* or the system script cisco. If there is no match for either the modem script or the system script, you will see a “no matching chat script found” message.

```
interface dialer 1
! v.32 rotaries are in rotary 1.
dialer rotary-group 1
! Use v.32 generic script.
dialer map ip 10.0.0.1 modem-script .*-v32 system-script cisco 1234
```

Modem-Specific Chat Script Examples

The following example shows line chat scripts being specified for lines connected to Telebit and US Robotics modems:

```
! Some lines have Telebit modems.
line 1 6
 script dialer telebit.*
! Some lines have US Robotics modems.
line 7 12
 script dialer usr.*
```

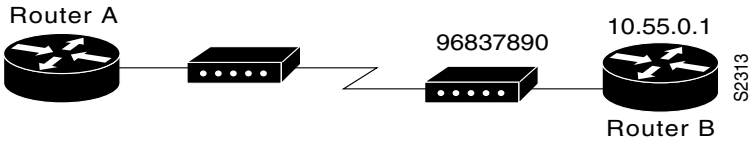
Dialer Mapping Example

The following example shows a modem chat script called dial and a system login chat script called login:

```
chat-script dial ABORT ERROR "" "AT Z" OK "ATDT \T" TIMEOUT 60 CONNECT \c
chat-script login ABORT invalid TIMEOUT 60 name: myname word: mypassword ">" "slip
default"
interface async 10
 dialer in-band
 dialer map ip 10.55.0.1 modem-script dial system-script login 96837890
```

Figure 1 illustrates the configuration.

Figure 1 Chat Script Configuration and Function



- The configuration is on Router A.
- The modem chat script dial is used to dial out to the modem at Router B.
- The system login chat script login is used to log in to Router B.
- The phone number is the number of the modem attached to Router B.
- The IP address in the **dialer map** command is the address of Router B.

In the sample script shown, the **dialer in-band** command enables DDR on asynchronous interface 10, and the **dialer map** command dials 96837890 after finding the specified dialing and the system login scripts. When a packet is received for 10.55.0.1, the first thing to happen is that the modem script is implemented. Table 3 lists the functions that are implemented with each expect-send pair in the modem script called dial.

Table 3 Example Modem Script Execution

Expect and Send Pair	Implementation
ABORT ERROR	Ends the script execution if the text “ERROR” is found. (You can have as many active abort entries as you like.)
“ ” “AT Z”	Without expecting anything, sends an “AT Z” command to the modem. (Note the use of quotation marks to allow a space in the send string.)
OK “ATDT \T	Waits to see “OK.” Sends “ATDT 96837890.”
TIMEOUT 60	Waits up to 60 seconds for next expect string.
CONNECT \c	Expects “connect,” but does not send anything. (Note that \c is effectively nothing; “ ” would have indicated nothing followed by a carriage return.)

After the modem script is successfully executed, the system login script is executed. Table 4 lists the functions that are executed with each expect-send pair in the system script called login.

Table 4 Example System Script Execution

Expect and Send Pair	Implementation
ABORT invalid	Ends the script execution if the message “invalid username or password” is displayed.
TIMEOUT 60	Waits up to 60 seconds.
name: *username*	Waits for “name:” and sends username. (Using just “name:” will help avoid any capitalization issues.)
word: *password*	Waits for “word:” and sends the password.
“>” “slip default”	Waits for the > prompt and places the line into Serial Line Internet Protocol (SLIP) mode with its default address.

System Login Scripts and Modem Script Examples

The following example shows the use of chat scripts implemented with the **system-script** and **modem-script** options of the **dialer map** command.

If there is traffic for IP address 10.2.3.4, the router will dial the 91800 number using the usrobotics-v32 script, matching the regular expression in the modem chat script. Then the router will run the unix-slip chat script as the system script to log in.

If there is traffic for 10.3.2.1, the router will dial 8899 using usrobotics-v32, matching both the modem script and modem chat script regular expressions. The router will then log in using the cisco-compressed script.

```
! Script for dialing a usr v.32 modem:
chat-script usrobotics-v32 ABORT ERROR "" "AT Z" OK "ATDT \T" TIMEOUT 60 CONNECT \c
!
! Script for logging into a UNIX system and starting up SLIP:
chat-script unix-slip ABORT invalid TIMEOUT 60 name: billw word: wewpass ">" "slip
default"
!
! Script for logging into a Cisco access server and starting up TCP header compression:
```

```
chat-script cisco-compressed...
!
line 15
 script dialer usrobotics-*
!
interface async 15
 dialer map ip 10.2.3.4 system-script *-v32 system-script cisco-compressed 91800
 dialer m
```