

# Implementing IOS CLI Menus

Michael J. Martin

As a postscript to the "terminal services" series that finished up last month, we will be looking at implementing IOS command line interface (CLI) menus. CLI menus enable network administrators to build overlay menu-driven interfaces for accessing EXEC shell commands. These provide the ability to create customized command interfaces that can allow simplified access for non-technical users or restrictive access for vendors and/or lower level technical support groups.

## Building Menus with the <menu> Command

The <menu> command exists as both an EXEC shell and configuration mode command. Menus are activated with the privilege level 1 EXEC command <menu {menu name}>. Menus are created using the global configuration command <menu {menu name} {subcommand/option}>. Here is a table of the menu configuration sub-commands and options:

Command	Function
menu <menu name>	This EXEC shell command starts the menu. Once active the menu cannot be terminated unless an "exit" method has been defined as a menu item, i.e., <exit>, <logout> or the <menu-exit> option.
menu <menu name> command {menu item key (# or Character)} {IOS EXEC Command}	This configuration mode command defines a menu item. These definitions are required for the menu to function.
menu <menu name> text {menu item key} {description}>	This configuration mode command defines the menu item description that is displayed when the menu is active. This is optional.
menu <menu name> options {pause   login}>	This configuration mode command defines optional behavior for when a menu item is activated. The pause option directs the command output into a text parser so the output can be read before returning the user to the menu interface. This is must when building menu items for EXEC shell commands. The login command requires user authentication before the menu item is executed. This is optional.
menu <menu name> title {text...}	This configuration mode command sets a global menu option that defines a title banner that is printed each time the menu is reloaded (which occurs after a menu item has been executed). This is optional.
menu <menu name> prompt {text...}	This configuration mode command sets a global menu option that defines the menu item descriptions that are displayed when the menu is loaded. This is optional.
menu <menu name> line mode	This configuration mode command sets a global menu option that requires a <cr> to follow each menu item. It is enabled automatically when a menu has more than nine definitions. Line mode allows you to backspace over line entries and use multiple character strings for menu item keys. This is an optional command with menus containing less than nine key definitions.
menu <menu name> single-space	This configuration mode command sets the menu display with single spaces.
menu <menu name> clear- screen	This configuration mode command sets a global menu option to redraw the menu after a menu item has executed.
menu <menu name> default {menu item}	This configuration mode command sets a global menu option that defines the menu item to be executed when a user sends a <cr> without

# Implementing IOS CLI Menus

Michael J. Martin

	a menu item.
menu <status-line>	This configuration mode command sets a global menu action that will display user status information.
menu <menu name> command {menu item key} menu-exit	This configuration mode command sets a global menu option that allows the user to exit the menu without ending the VTY session. It is used in nested menu configurations when a user needs to exit a submenu and return to the main menu. If no "higher" menu exists, it returns the user to the EXEC shell.

The anatomy of a menu consists of four parts:

```
<-----HEADER----->
#####
##                                     ##
##          Welcome to Outlan Terminal Services          ##
##                                     ##
#####

<-----MENU ITEMS -> MENU KEYS----->

1.          Connect to outlan-gw
2.          Connect to outlan-rt02
3.          Connect to outlan-sw01
4.          Connect to outlan-modem

<-----FOOTER----->
```

It is possible to have more than one session open. Once a session has been started you may return to this menu and open another. To leave a session type "ctrl+shift+6" then "x". You will then be returned to this menu.

To see open sessions type: "l"  
To disconnect from a session type: "d"  
To resume a sessions type: "r"  
To logoff type: "q"

Type Your Selection:

- Menu-items are the commands available in the menu. These can be either IOS commands or menu options. The menu items correspond to menu-keys that are either numeric or text strings that the user enters to execute the command.
- Menu-keys are the text statements displayed to the user that inform the user of the numeric or text key to activate the menu item. Menu-keys are defined using the menu text option, which sets them in the middle of the menu screen. However, menu key definitions can also be defined in the menu header or footer.
- Menu-title (header) is the text field at the top of the menu that can contain instructions for the user on the commands available and usage instructions.

# Implementing IOS CLI Menus

Michael J. Martin

- Menu-prompt (footer) is the text field at the bottom of the menu that can contain instructions for the user on the commands available and usage instructions.

Technically, of the four menu parts, only menu-items need to be defined for the menu EXEC command to function, although, without any key, header, or footer information, the menu is only a useless blank screen. As far as a menu structure is concerned, each menu is limited to 18 menu items. However, it is possible to nest menus (have one menu call another menu); the depth of nested menus varies in each IOS version. Nesting more than three deep is a safe bet. In the event that IOS has a problem with nesting, it will stay in the current menu.

## Using Menus

The goal of using menus is to supplant the EXEC shell with either a more user friendly or restrictive (depending on your motivation) interface. The menu interface is activated in the EXEC shell and then runs on top of it. Commands called through the menu interface must behave in correspondence to the privilege level in which the menu was activated. Depending on the commands you want to run through the menu, you may need to adjust the privilege levels of commands. This is an important fact to keep in mind when setting up how your menu executes. Two approaches that can be followed: local menu execution and remote menu execution.

The local approach is quick, simple, and limiting. It involves configuring the local VTY lines to run <autocommand> upon a successful login. Here is an IOS syntax example that configures the router's first four VTY lines to run the menu named "ts-main":

```
line vty 0 4
Privilege level 5
autocommand menu ts-main
transport input telnet
```

The limiting aspect of this approach is that it "dedicates" the VTY lines to "menu" access only. To provide EXEC shell management access, additional VTY lines need to be configured and assigned to a rotary group. Here is a syntax example that assigns VTY lines 5 through 8 to rotary group 1, making them accessible by opening a telnet session to port 3001:

```
Line vty 5 8
rotary 1
transport input telnet
```

While the local approach is simple and limited, remote is more involved and flexible. Remote execution depends on AAA and TACACS+ EXEC authorization support instead of configuring <autocommand> to execute the menu directly on the VTY line. A user or group profile on TACACS+ server is configured to execute the menu command once the user has authenticated. Here is a syntax example for the tac-plus TACACS+ implementation:

```
user = menutest {
    login = cleartext "emmetis0ne"
    member = admin
service = exec {
    priv-lvl = 15
    autocmd = "menu mgt"
}
}
```

# Implementing IOS CLI Menus

Michael J. Martin

While some additional configuration is needed for this approach, using TACACS+ to manage menu execution has the advantage of allowing the VTY lines to be left open for EXEC shell access. TACACS+ user profiles determine how the user's session will operate. This makes it possible to implement support for different menu interfaces and session behaviors, resulting in highly customized user/group menu interfaces. Remember, from a security perspective users only need access to functionality that allows them to accomplish their tasks. Anything more increases risk, and the networking environment is one arena where you want to minimize risk as much as possible.

## Configuring Menus

Those of you who have read my terminal server series of articles should at this point have a pretty deep understanding of how to access both the router's CLI and those devices connected to its TTY serial lines. That said, some of you are probably of the mind that while this access and control method does work, it leaves a lot to be desired for users who are not familiar with the IOS or the idiosyncrasies of TTY lines. For this reason, we will look at two configuration examples, one implementing a terminal server interface, and the other providing a support interface.

Menus, like standard and extended access lists are not editable. If one entry is deleted, the whole list is deleted. Menus also cannot be deleted if in use, but they can be modified. The operational considerations make it a good idea to construct menus offline and then load them either by pasting the configuration into an active configuration session or using the <copy> command.

Lets start with the support interface. In large environments, it is not uncommon to utilize a tiered support model. This menu provides first-level support personal access to critical monitor data. It provides access to ping, Traceroute, IP routing tables, and ARP, NAT, CBAC and process tables. Here is the menu interface displayed to the user:

```
#####  
##                                                                 ##  
##           Welcome outland-gw monitoring services           ##  
##                                                                 ##  
#####  
  
2.           Ping Command  
3.           Traceroute Command  
4.           IP routing Table  
5.           ARP Table  
6.           NAT Table  
7.           Show Interfaces  
8.           CBAC Sessions  
9.           Show Processes
```

Welcome to the Level One Management Interface! Type the number of the command you wish to run, then follow the corresponding prompts.

2=PING, 3=Traceroute, 4=IP Table, 5=ARP Table, 6=NAT Table  
7=Interface Status, 8=CBAC Sessions, 9=Processes, q=Exit

Command:

Notice that the menu keys are as I mentioned earlier are in list form and in the footer of the menu. The configuration is as follows:

# Implementing IOS CLI Menus

Michael J. Martin

The menu title (header), set with <menu {menu name} title {text...}>:

```
menu mgt title ^
#####
##                                     ##
##           Welcome to outland-gw monitoring services           ##
##                                     ##
#####
^
```

The menu prompt (footer), set with <menu {menu name} prompt {text...}>:

```
menu mgt prompt ^
Welcome to the Level One Management Interface! Type the number of
the command you wish to run, then follow the corresponding prompts.
```

```
2=PING, 3=Traceroute, 4=IP Table, 5=ARP Table, 6=NAT Table
7=Interface Status, 8=CBAC Sessions, 9=Processes, q=Exit
```

Command: ^

Menu key statements, set with <menu {menu name} text {menu key text...}>:

```
menu mgt text 2. Ping Command
menu mgt text 3. Traceroute Command
menu mgt text 4. IP routing Table
menu mgt text 5. ARP Table
menu mgt text 6. NAT Table
menu mgt text 7. Show Interfaces
menu mgt text 8. CBAC Sessions
menu mgt text 9. Show Processes
```

Menu item statements, set with <menu {menu name} command {command}>:

```
menu mgt command 2 ping
menu mgt command 3 traceroute
menu mgt command 4 sh ip route
menu mgt command 5 sh arp
menu mgt command 6 sh ip nat translations
menu mgt command 7 sh ip interface brief
menu mgt command 8 sh ip inspect sessions
menu mgt command 9 sh processes
menu mgt command q menu-exit
```

Menu option statements, set with <menu {menu name} options {menu key text...} pause>:

```
menu mgt options 2 pause
menu mgt options 3 pause
menu mgt options 4 pause
menu mgt options 5 pause
menu mgt options 6 pause
menu mgt options 7 pause
menu mgt options 8 pause
menu mgt options 9 pause
```

# Implementing IOS CLI Menus

Michael J. Martin

```
menu mgt options q login
Menu clear-screen option, set with <menu {menu name} clear-screen>:
menu mgt clear-screen
```

The terminal server interface follows a slightly different format. Commands to open sessions to the connected devices are listed using menu items statements. The connection management commands, however, are listed in the menu footer. The menu provides access to the connected devices and access to the <show sessions>, <resume> and <disconnect> commands. Here is the menu as it appears to the user after login:

```
#####
##                                     ##
##           Welcome to Outlan Terminal Services           ##
##                                     ##
#####

1.          Connect to outlan-gw
2.          Connect to outlan-rt02
3.          Connect to outlan-sw01
4.          Connect to outlan-modem
```

It is possible to have more than one session open. Once a session has been started you may return to this menu and open another. To leave a session type "ctrl+shift+6" then "x". You will then be returned to this menu.

```
To see open sessions type: "l"
To disconnect from a session type: "d"
To resume a sessions type: "r"
To logoff type: "q"
```

Type Your Selection:

This terminal server menu configuration is intended to be an example from which to build. In a typical terminal server environment, a nested, multi-menu architecture would need to be implemented to accommodate both the potential number of sessions (between 16 and 128) and the session management commands. Here is the menu configuration syntax broken down by section:

The menu title (header), set with <menu {menu name} title {text...}>:

```
menu tsa title ^
#####
##                                     ##
##           Welcome to Outlan Terminal Services           ##
##                                     ##
#####
^
```

The menu prompt (footer), set with <menu {menu name} prompt {text...}>:

```
menu tsa prompt ^
```

# Implementing IOS CLI Menus

Michael J. Martin

It is possible to have more than one session open. Once a session has been started you may return to this menu and open another. To leave a session type "ctrl+shift+6" then "x". You will then be returned to this menu.

To see open session's type: "l"  
To disconnect from a session type: "d"  
To resume a session's type: "r"  
To logoff type: "q"

Type Your Selection: ^

Menu key statements, set with <menu {menu name} text {menu key text...}>:

```
menu tsa command 1 telnet outlan-gw
menu tsa command 2 telnet outlan-rt02
menu tsa command 3 telnet outlan-sw01
menu tsa command 4 telnet outlan-modem
menu tsa command l show sessions
menu tsa command d disconnect
menu tsa command r resume /next
menu tsa command e menu-exit
menu tsa command q exit
```

Menu item statements, set with <menu {menu name} command {command}>:

```
menu tsa text 1. Connect to outlan-gw
menu tsa text 2. Connect to outlan-rt02
menu tsa text 3. Connect to outlan-sw01
menu tsa text 4. Connect to outlan-modem
```

Menu option statements, set with <menu {menu name} options {menu key text...} pause>:

```
menu tsa options 1 pause
menu tsa options 2 pause
menu tsa options 3 pause
menu tsa options 4 pause
menu tsa options l pause
menu tsa options d pause
menu tsa options r pause
```

Menu clear-screen option, set with <menu {menu name} clear-screen>:

```
menu tsa clear-screen
```

That wraps up our coverage of IOS CLI menus. I hope that you have found this tip helpful and applicable to your network environment.