

# Conveniently Administering Remote Servers

by Boris Loza, PhD, CISSP

Imagine that you're in charge of 40 Solaris/SunOS servers on a 24x7 basis. To make life easier for you, let's suppose that some of these servers are located a good two-hour drive from your house, and some are even a three-hour flight away.

Then one night, when you're ready to go to sleep (or even already dreaming), your pager starts beeping. One of the servers is down. After trying to `telnet` and `ping` to this server several times, you suddenly realize that something is really wrong. What are you going to do?

You have two options. First, you start packing your bags. Second, you stay at home, and from the server's console, bring the system to the OpenBoot `ok` prompt. After investigating a boot disk failure, you then boot from the alternative hard drive. With that done, you go back to sleep.

In this article, we'll show you how you can take advantage of the second option. First, though, we have to collect some equipment.

## Equipment

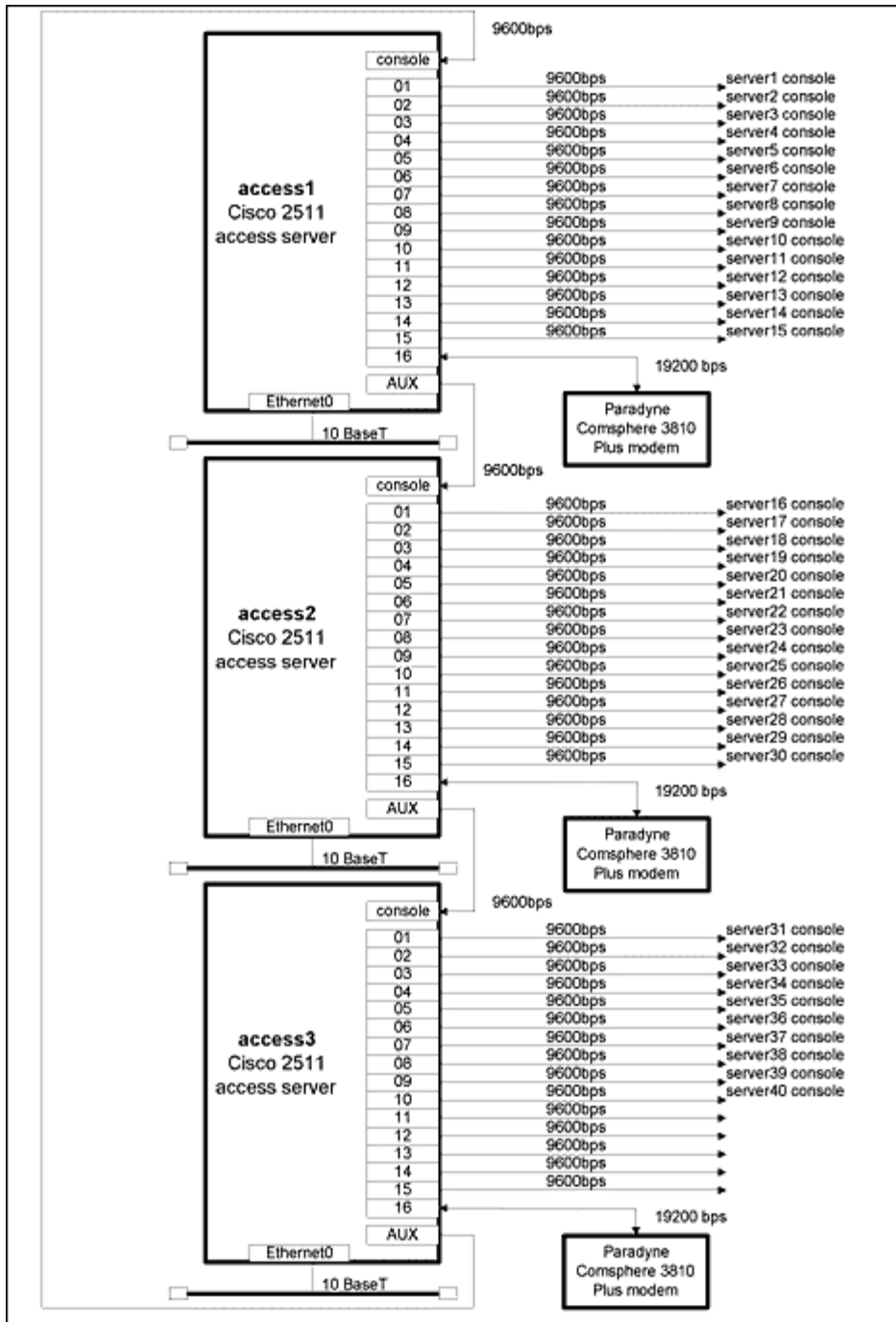
In order to dial up to the console port on the UNIX box, we first have to wire it to a terminal server. To do so, we'll use Cisco 2509 or Cisco 2511 access servers. Both of these servers include 1 Ethernet port, 2 synchronous serial ports, and 8 (Cisco 2509) or 16 (Cisco 2511) asynchronous serial ports. We also need a modem with good network security. The Paradyne Comsphere 3800 Plus Series modem works well. These modems have a user-friendly configuration, support up to 33.6 KB, and provide sophisticated network security. The next step is to hook all of your equipment together.

## Connection

Because the access server has only 16 asynchronous serial ports (Cisco 2511), and you're in charge of 40 boxes, we'll use 3 Cisco 2511 (or 6 Cisco 2509) terminal servers. (The terms access and terminal servers have the same meaning and will be used interchangeably in this article.)

You'll see a connection diagram shown in **Figure A**.

**Figure A:** *This is the schematic of our access server communications.*



As you can see from this diagram, we've connected all the Cisco servers together through an auxiliary port. Ports 1 through 15 are connected to the servers' console ports using octal cables. We configured these ports to support 9600 bps. Port 16 on each Cisco server is connected to the Paradyne Comsphere 3810 Plus modem at 19200 bps. All access servers are connected to our local Ethernet network through a 10Base-T Ethernet port. Because we'll dial into the private network, it's crucial to achieve maximum security.

## Security

Our configuration provides four levels of security. The first level is an eight-digit numeric DIAL-IN password on the modem. The password is transmitted to the modem in the form of DTMF (Dual Tone Multiplexed Frequency) tones. After dialing the modem, it generates an artificial dial tone until the password is correctly entered. You can only change the password from the front panel of the modem. Callback security is also provided.

In the second level, we have to go through EXEC username and password on the access server. This is required to log on to the access server from the modem or the network. You must do this before a PPP session can be started.

The third level requires a SERIAL LINE password. This is an alphanumeric password that's set up on the access server for each of the asynchronous serial lines (with the exception of the modem line). The password is required every time a user attempts to access the console port on a UNIX server or the console port on one of the terminal servers.

Finally, the fourth level requires a username and password on the UNIX console. When configuring the access server, an ENABLE secret password is required. This is equivalent to the root password on a UNIX box. You reach enable mode by typing `enable` and entering `enable's` password.

## Configuration

The access server and modem configuration are beyond the scope of this article. For instructions on how to set up asynchronous serial lines, passwords, and PPP on a terminal server, consult Cisco's technical documentation. For modem configuration, consult the technical manual for your modem.

You can gain dial-up access into a terminal server from any computer equipped with terminal emulation software and a modem. Just set the terminal emulator to 8 bits, 1 stop-bit, and no parity. The modem password must be embedded in the dial string. It should also be separated from the phone number by a series of commas.

On most modems, a comma defaults to a two-second pause. Typically, three or four commas are required. You must terminate the password with a pound sign (#)—for example

```
4163331234,,,09876543#
```

In this example, 4163331234 is a phone number and 09876543 is a modem password. Note that some terminal emulators may treat the pound sign as a comment. Consult the documentation that came with your terminal emulator for further information.

The terminal servers are configured to use proxy-arp. The IP address, which is assigned to the dial-up computer, belongs to the same subnet as the access server's own IP address. The access server will respond to an arp request on behalf of the dial-up computer. It will also take care of forwarding packets between the Ethernet and the dial-up computer. The network routers don't need to know that the dial-up computer isn't physically attached to the Ethernet segment.

The PPP software on the dial-up computer should be configured so that it doesn't provide an IP address for either end of the link and accept the IP address assigned by the terminal server. It should also use a maximum packet length (MTU) of 1500 and should not attempt to use PAP or CHAP. A typical chat script is shown on **Listing A**.

**Listing A:** *Typical modem's chat script*

```
ABORT "NO CARRIER"
ABORT BUSY
" " ATDT${PHONE} , , ,
${NAMERIC PASSWORD}#
"Username: " ${USERNAME}
>Password: " ${PASSWORD}
">" ppp
```

Asynchronous lines 1-15 on each terminal server are wired to the console ports on the UNIX servers. So, if you're located anywhere on your private network, you can establish virtual connections to the UNIX console ports using `telnet`. This process is referred to as reverse-telneting.

Any specific asynchronous line on a terminal server can be accessed by telneting to the server using a TCP port number, which is calculated as follows:

```
TCP port number = 2000 + n
```

where `n` corresponds to the required line number. For example, line 5 of a terminal server is accessed by telneting to the terminal server on TCP port 2005.

Keeping track of the physical connections between the terminal servers and the UNIX boxes is a difficult task. For this reason, it's worthwhile to set up aliases for IP addresses on a terminal server. It's also worthwhile to set up aliases for each of the accessible serial lines. After this has been done, you can telnet directly to a specific serial line using TCP port 23. After you connect to the terminal server (by dialing up or telneting), you're prompted for a login.

## Session

You must enter a valid username and password in order to log on to the access server. After logging on, the terminal server generates the Cisco `exec` prompt. As shown in **Listing B**, a user establishes a virtual connection to the console port on the UNIX server, logs on to the console, determines who's logged on, halts the machine, runs the banner command, resumes the halt, and terminates the session.

As you can see from **Listing B**, you can set a break (same as using `STOP A` on the Solaris keyboard) by terminating a reverse-telnet session with `^]` and typing `send break` at the telnet prompt. Now the terminal from which you're running this session becomes a console. It's like you're sitting at a console terminal connected directly to your SPARC machine.

### **Listing B:** *A session example with the cisco access server*

```
$telnet server1_console
Cisco Access Server (ts)
User Access Verification
Username: user_adm
Password: *****
Password OK
SERIAL LINE PASSWORD ACCEPTED
server1 console login: root
password: *****
$stty
/dev/console
$who
user1 console Jun  5 14:26
$exit
server1 console login:
^]
telnet> send break
ok
ok banner
SPARCstation 2, Type 4 Keyboard
ROM Rev. 2.0, 16MB memory installed, Serial # 289
Ethernet address 8:0:20:d:e2:7b, Host ID: 55000121
ok go
telnet> q
Connection Closed
```

After completing a login session on a console, it's necessary to type `exit` (or `^D`) before terminating the virtual console connection. Failure to do this (also due to a network error or PC crash) will cause the login session to remain active even after the connection to the console port has been terminated.

To clear an established connection on a serial line, you can use the `clear line n` command from the `exec` prompt (where `n` is the serial line number that you would like to clear). Be careful

not to clear a line someone else is actually using. Always type the `who` or `show users` command at the `exec` prompt first.

You can also use reverse-telnet to access modems from anywhere on the enterprise network. Once you establish a reverse-telnet session, you can send `AT` commands to a modem. To configure and reset the modem, dial a remote computer, or dial a pager. Note that you can't change the modem password using `AT` commands.

The modems are connected to asynchronous line 16 on the access servers. You can establish connections to the modems by telnetting to TCP port 2016 on the terminal servers or by telnetting to TCP port 23 using the aliases for the IP address. In **Listing C**, you can see how the modem on an access server can be reset.

**Listing C:** *Reset the modem on a terminal server*

```
$telnet server1 2016
Cisco Access Server (ts)
User Access Verification
Username: user_admin
Password: *****
Password OK
SERIAL LINE PASSWORD ACCEPTED
atz
OK
^]
telnet> q
Connection closed
```

The auxiliary asynchronous lines and the console ports on the access servers have been wired together in a circular fashion, as shown in **Figure A**. You can access console ports using the same steps used to access the UNIX console ports. The auxiliary line is the 17th line available on the terminal servers, and therefore, the TCP port number required is  $2000 + 17 = 2017$ . You can set up an IP alias for the auxiliary line on each communication server and access the terminal server console port by telnetting to TCP port 23. Access to the terminal server's console port is useful when performing an IOS reload (reboot) or when a network failure has occurred.

## Conclusion

In this article, we've shown you that you can easily administer a remote Solaris/SunOS system while still being secure. If you're responsible for a number of UNIX servers on a 24x7 basis, you can solve most problems by simply dialing in or telnetting to the server's console from home or from your private network. You can access the server even in the event of a hardware or network failure. This can make life a little bit easier for UNIX System Administrators!