International School

Capstone Project 2

CMU-CS451

IMPLEMENT DOCUMENT

Version 1.0

Date: 15/02/2022

IMPLEMENT WEB HOSTING SYSTEM USING HAPROXY TO PERFORM LOAD BALANCING SERVICE

Submitted by : Vien, Nguyen Le

Approved by : MSc. Thach, Tran Ban

Proposal Review Panel Representative:

Name	Signature	Date
Nguyen Le Vien		

Capstone Project 2- Mentor:

Name	Signature	Date
MSc. Thach, Tran Ban		

PROJECT INFORMATION

Project Title	Implement Web hosting system using HAproxy to perform load balancing service		
Start Date	17 Feb 2022	**End Date**	9 May 2022
Lead Institution	International School, Duy Tan University		
Project Mentor	MSC. Thach Tran Ba Email: tranbanthach@gmail.com Tel: 0931270979		
Scrum master / Project Leader & contact details	Vien, Nguyen Le Email: viennguyen110194@gmail.com Tel: 0348037953		
Team members	Name	Email	Tel
	Vien, Nguyen Le	viennguyen110194@gmail.com	0348037953

REVISION HISTORY

Version	Date	Comments	Author	Approval
1.0	17 Feb	Initial Release	C2NE.03 Team	

TABLE OF CONTENTS

LIST OF TABLE

IST OF FIGURE

1. INTRODUCTION

1.1. Purpose

C2 Deployment for Pentesting is an authoritative attack on a computer system, performed to evaluate the security of the system/network. This test is performed to determine how malicious code is stored on the enterprise's system and deploys SIEM applications to perform remediation and provide solutions to respond to attacks. The first step is to install the Cobalt Strike toolkit to support Pentesting. The second step should provide general details on how to install the SIEM toolkit on the terminals.

2. IMPLEMENT

2.1. How to setup Cobal Strike?

Before installing Cobalt Strike, please make sure that you have Oracle Java installed with version 1.7 or above. You can check whether or not you have Java installed by executing the following command:Li

```
┌──(kali㉿kali)-[~/Desktop]
└─$ java --version
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -
Dswing.aatext=true
openjdk 11.0.16 2022-07-19
OpenJDK Runtime Environment (build 11.0.16+8-post-Debian-1)
OpenJDK Server VM (build 11.0.16+8-post-Debian-1, mixed mode, sharing)
```

Table 3.1 : Check If the java environment

If you receive the java command not found error or another related error, then you need to install Java on your system. You can download this here: https://www.java.com/en/

Cobalt Strike comes in a package that consists of a client and server files. To start with the setup, we need to run the team server. The following are the files that you'll get once you download the package:

```
┌──(kali㉿kali)-[~/Desktop/CS4.5]
└─$ ls -l
total 33772
-rwxrwxrwx 1 kali kali      126 Dec  5  2019 agscript
-rwxrwxrwx 1 kali kali      144 Dec  5  2019 c2lint
-rwxrwxrwx 1 kali kali      256 Jan 11  2022 cobaltstrike.auth
-rwxrwxrwx 1 kali kali 34150633 Jan 11  2022 cobaltstrike.jar
-rwxrwxrwx 1 kali kali     2741 Feb 10  2022 cobaltstrike.store
drwxr-xr-x 2 kali kali     4096 Nov  4 05:34 data
-rwxrwxrwx 1 kali kali    96139 Feb 10  2022 icon.jpg
drwxr-xr-x 7 kali kali     4096 Nov 14 01:51 logs
-rwxrwxrwx 1 kali kali      141 Dec  5  2019 peclone
```

```
-rwxrwxrwx 1 kali kali      553 Feb 28  2022 README.md
-rwxrwxrwx 1 kali kali       89 Dec 22  2021 start.bat
-rwxrwxrwx 1 kali kali      108 Dec 30  2021 start.sh
-rwxrwxrwx 1 kali kali     1877 Feb 27  2022 teamserver
-rwxrwxrwx 1 kali kali     2017 Feb 10  2022 teamserver.bat
drwxr-xr-x 2 kali kali     4096 Feb 28  2022 third-party
-rwxrwxrwx 1 kali kali       87 Dec  5  2019 update
-rwxrwxrwx 1 kali kali   271466 Dec  5  2019 update.jar
```

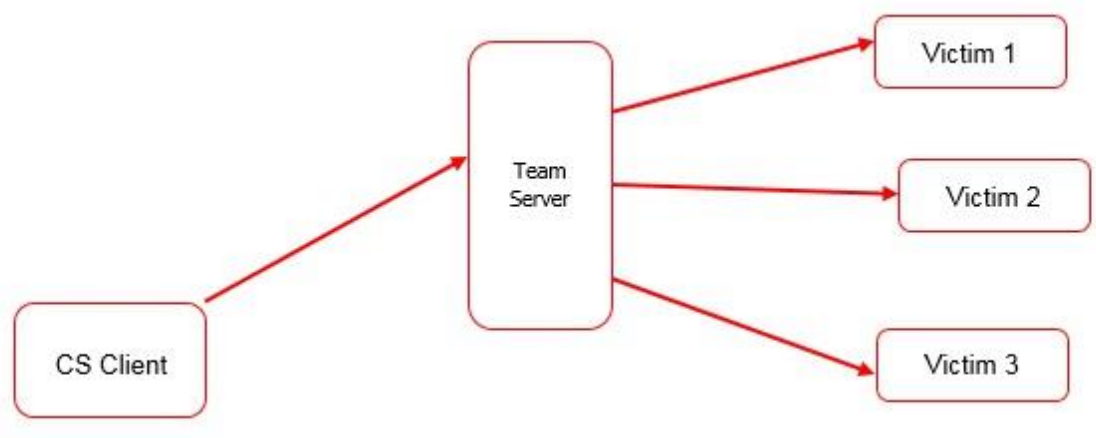
Table 3.2 : List and check files

The first thing we need to do is run the team server script located in the same directory.

2.2. What is a team server ?

- This is the main controller for the payloads that are used in Cobalt Strike.
- It logs all of the events that occur in Cobalt Strike.
- It collects all the credentials that are discovered in the post-exploitation phase or used by the attacker on the target systems to log in.
- It is a simple bash script that calls for the Metasploit RPC service (msfrpcd) and starts the server with cobaltstrike.jar. This script can be customized according to the needs.

Cobalt Strike works on a client-server model in which the red-teamer connects to the team server via the Cobalt Strike client. All the connections (bind/reverse) to/from the victims are managed by the team server.



As shown in the following screenshot, the team server needs at least two mandatory arguments in order to run. This includes **host**, which is an IP address that is reachable from the internet. If behind a home router, you can port forward the listener's port on the router. The second mandatory argument is **password**, which will be used by the team server for authentication:

```
┌──(kali㉿kali)-[~/Desktop/CS4.5]
└─$ sudo ./teamserver
[sudo] password for kali:
[*] Will use existing X509 certificate and keystore (for SSL)
[*] ./teamserver <host> <password> [/path/to/c2.profile] [YYYY-MM-DD]

      <host> is the (default) IP address of this Cobalt Strike team server
      <password> is the shared password to connect to this server
      [/path/to/c2.profile] is your Malleable C2 profile
      [YYYY-MM-DD] is a kill date for Beacon payloads run from this server
```

Table 3.3 : Run Teamserver

```
┌──(kali㉿kali)-[~/Desktop/CS4.5]
└─$ sudo ./teamserver 192.168.119.150 pass
```

Table 3.4 : Commands run correctly

Here, I am using the IP 192.168.119.150 as my team server and pass as my password for the team server:

```
┌──(kali㉿kali)-[~/Desktop/CS4.5]
└─$ sudo ./teamserver 192.168.119.150 pass
[sudo] password for kali:
[*] Will use existing X509 certificate and keystore (for SSL)
[-] Trapped java.net.BindException during team server startup [main]: Address already
in use (Bind failed)
java.net.BindException: Address already in use (Bind failed)
      at java.base/java.net.PlainSocketImpl.socketBind(Native Method)
      at
java.base/java.net.AbstractPlainSocketImpl.bind(AbstractPlainSocketImpl.java:452)
      at java.base/java.net.ServerSocket.bind(ServerSocket.java:395)
      at java.base/java.net.ServerSocket.<init>(ServerSocket.java:257)
      at java.base/javax.net.ssl.SSLServerSocket.<init>(SSLServerSocket.java:181)
      at
java.base/sun.security.ssl.SSLServerSocketImpl.<init>(SSLServerSocketImpl.java:78)
      at
java.base/sun.security.ssl.SSLServerSocketFactoryImpl.createServerSocket(SSLServe
rSocketFactoryImpl.java:87)
      at ssl.SecureServerSocket.<init>(Unknown Source)
      at server.TeamServer.A(TeamServer.java:71)
      at server.TeamServer.main(TeamServer.java:118)

┌──(kali㉿kali)-[~/Desktop/CS4.5]
└─$ ./start.sh
```

Table 3.5 : Showing results

Upon successfully starting the server, we can now get on with the client. To run the client, use the following command:

```
java -jar cobaltstrike.jar
```

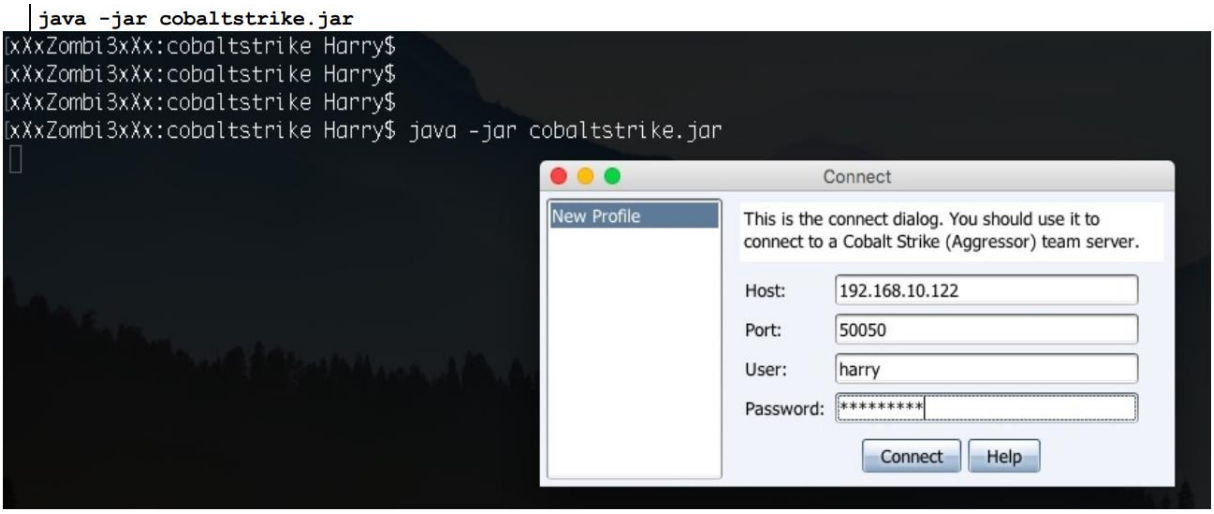


Figure 1.1: Teamserver connect

2.3. Basic configuration SIEM application

To install Splunk on Ubuntu, the developers of this platform offers Deb binary that easily can be downloaded from the https://www.splunk.com.

Alternatively, the users can use the below given **wget** command to get the free version of Splunk with trial Enterprise features.

```
wget -O splunk-8.2.1-ddff1c41e5cf-linux-2.6-amd64.deb
'https://www.splunk.com/page/download_track?file=8.2.1/linux/splunk-8.2.1-ddff1c41e5cf-linux-2.6-
amd64.deb&ac=&wget=true&name=wget&platform=Linux&architecture=x86_64&version=
8.2.1&product=splunk&typed=release"
```

Table 3.6 : Download splunk

As the downloaded file is .deb, thus we can use the APT package manager to install it.

Note: If you have downloaded this data analyses software on GUI Linux using the browser, the first switch to the Downloads directory using cd Downloads. Whereas the users got it using wget command can simply run:

```
sudo apt install ./splunk-*-amd64.deb
```

Table 3.7: Command to install Splunk on Ubuntu 20.04

Once the installation is completed, let's run the script that will not only enable Splunk service at boot level but also let us set up login details- **Admin** user and its **password**. However, as the script starts press the **Esc** key and the **Y** to accept the license.

sudo /opt/splunk/bin/splunk enable boot-start

Table 3.8: Run Start Splunk

Now, this data analytical platform is ready, lets's access its web interface at **localhost:8000**, whereas the users who want to access Splunk Dashboard on some remote system, need to open port **8000** in the system firewall. For that run:

sudo ufw allow 8000

Table 3.9: Open port 8000 in system firewall

Result:

For remote system browser : - http://your-server-ip:8000

For Local system browser : - http://localhost:8000

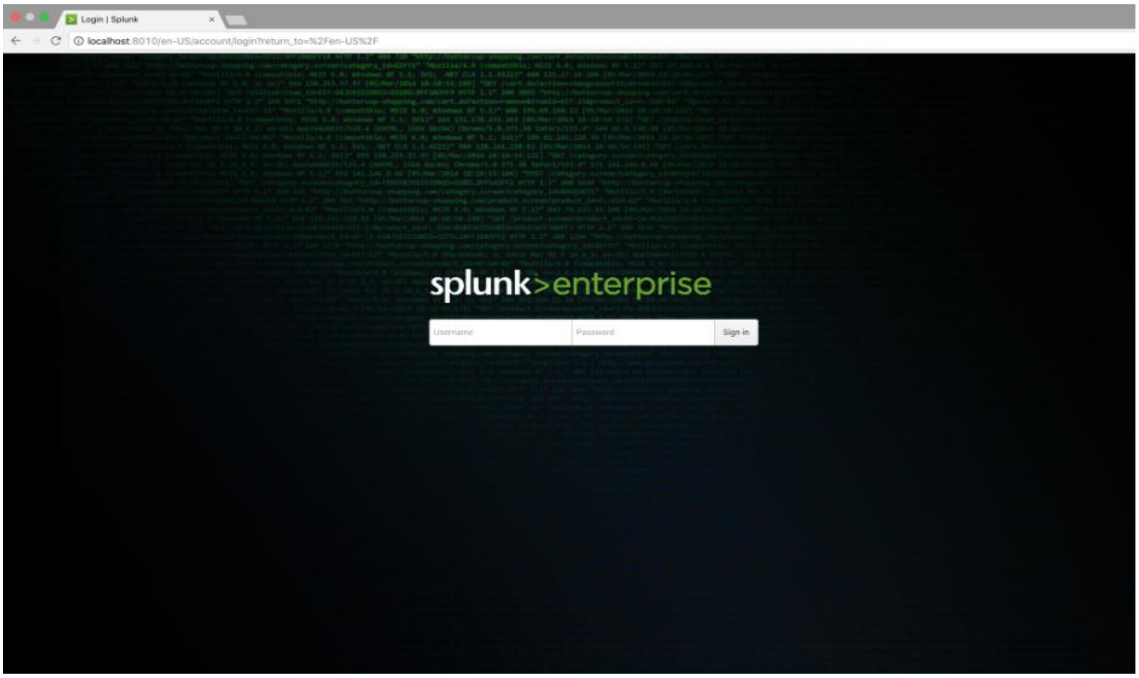


Figure 1.2: Admin interface

2.4. Config Log on Endpoint

You need to go the homepage https://www.splunk.com and install Splunk Universal Forwarder. If you need the full steps, you can refer to the link https://www.youtube.com/watch?v=g-NC2GVhzec

```
# cd /opt/splunk/etc/deployment-apps/
# mkdir Windows
# cd Windows
# touch inputs.conf
# touch ouputs.conf
# vim inputs.conf
```

Table 1.3 : Create file config

```
[default]
index = win10log

[WinEventLog://Security]
disable = 0
current_only = 1

[WinEventLog://System]
disable = 0
current_only = 1

[WinEventLog://Application]
disable = 0
current_only = 1
```

Table 1.4: Open file inputs.conf

```
[tcpout]
defaultGroup = default-autolb-group

[tcpout:default-autolb-group]
server = My_ip_Server:9997

tcpout-server://My_ip_Server:9997
```

Table 1.5: Open file ouputs.conf

After configuring everything we need to reload the app SIEM.

```
# /opt/splunk/bin/splunk reload deploy-server
```

Table 1.6: Reload Splunk

3. Test Result

3.1. Cobalt Strike Result

First, start the Cobalt Strike team server and connect to it. Once we have the interface up and running, we will start a listener. A listener is a handler that handles all the incoming connections. To do this, go to the Cobalt Strike menu and choose Listeners, as shown in the following image:

Figure 1.3: Create Listensers

This will open

3.2.