Project – Blue / Red
Team Game Report 2

CSCI 6706 Network Design

Christian Liu – B00415613
Computer Science Department
Dalhousie University
Halifax, NS
Chris.liu@dal.ca

Table of Contents

ABSTRACT

The second term project report will cover following materials [2]:

- Blue / Red team software installation and configuration.
- Penetration testing session Blue /Red team plans.

KEYWORDS

Network Security, Red team, Blue team, Penetration testing, Metasploit, DVWA

1 Introduction

The second report is refinement of the first report [5]. It does not have much adjustment based on the plan of the first report, except Privilege Escalation consideration. Moreover, it added fine-grinded details about how and what software install and configured for Blue / Red team, Plus, I planned penetration testing session for Blue /Red team in this report. I upgraded Blue / Red Team software list as below:

- **Software prepared on both Blue and Red Team**

Blue Team	Red Team
WireShark (Traffic Monitoring, Sniffing, and recording)	Metasploit Framework
Argue (Data Capturing, Converting and Analyzing)	Hpring3 (DoS /DDoS)
TCPDump (Data Sniffing)	Nmap (Port Scan)
TCPReplay / TCPRewrite (Traffic replay / rewriting)	Medusa (Brute-Force)
ACL (access-control)	Hydra (Brute-Force)
UFW (firewall)	Patator (Brute-Force)
Whitelist / Blocklist (Filter)	ZAP (Web Vulnerability)
Snort (Traffic Monitoring and Alerting)	DirtyCow (Privilege Escalation)
Blockchain (tentative – Personal recommendation)	

- **Penetration Testing**

In term of Penetration Testing, I am planning to include "Privilege Escalation".

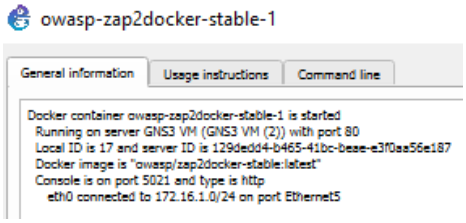
ICMP / SYN Flood	Hping3
Nmap	Metaploit Framework (Port Scan)
Brute-Force	Medusa, Hydra and Patator
Privilege Escalation	Metaploit Framework (Dirty Cow)
Web Vulnerability	ZAP or SQL injection or manually

2 Software Installation and Configuration

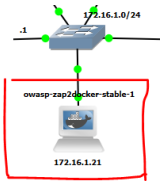
2.1 Red team software

2.1.1 OWASP ZAP

OWASP ZAP (short for Zed Attack Proxy) is an open-source web application security scanner. It is intended to be used by both those new to application security as well as professional penetration testers [7].
Installation on GNS3 VM is quite straightforward "**sudo snap install zaproxy –classic**". Alternatively, we can create the alliance template via GNS3, which is Docker type:



Add ZAP Docker image in current topology:



2.1.1 Metasploit Framework

Metasploit Framework accumulates bunch of penetration tools together. And it is easily to be managed by developer / tester. It can be install through https://www.offensive-security.com/. After installation we can initialize the Metasploit DB and add / use the modules, for example: nmap, dirty cow etc.

2.1.2 DirtyCow – Privilege Escalation

Download Dirty Cow source code [11], then compile it into executable file:
$ gcc -pthread -lcrypt dirty_cow.c -o dirty_cow

2.1.3 Hping3

Hping3 is a command-line oriented TCP/IP packet assembler/analyzer. The interface is inspired to the ping(8) Unix command, but Hping3 isn't only able to send ICMP echo requests. It supports TCP, UDP, ICMP and RAW-IP protocols, has a traceroute mode, the ability to send files between a covered channel, and many other features [9]. We can easily install it on GNS3 VM via "apt-get install" command.

2.1.4 Nmap

Till now, we already setup snort configuration for tuning up. Next, we want to test our real datasets downloaded from DARPA.
Nmap can be easily installed as "sudo apt-get install nmap" on GNS3 VM. It can be used to conduct port scan attack by options.
sudo nmap -sT -p1000-2000 172.16.1.20

2.1.5 Medusa, Hydra and Patator

I am going to focus on tools that allow remote service brute-forcing. These are typically Internet facing services that are accessible from anywhere in the world. The three tools I will assess are Hydra, Medusa and Patator (from nmap.org). Installation of all three tools was straight forward on Ubuntu Linux. Use the standard method to compile an application from source.

- wget https://raw.githubusercontent.com/lanjelot/patator/master/patator.py
- wget https://github.com/vanhauser-thc/thc-hydra/archive/v9.0.tar.gz
- wget http://www.foofus.net/jmk/tools/medusa-2.2.tar.gz

2.2 Blue team software

Because I am working on Windows platform, PRTG seems to be more friendly to me, compared with Cacti. Therefore, to answer the Question 1, I was mainly using PRTG tools to constantly monitor the devices over my network during several

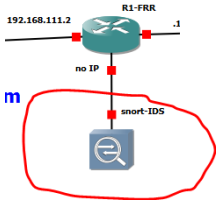
2.1.1 Wireshark

Wireshark can be easily installed on GNS3 VM by executing the command "sudo **apt-get install** libcap2-bin **wireshark".** And it had been ready since last assignments. The configuration and customization were mentioned in my assignment 2 specifically[3**].**

dumpcap -I 3 -q -b duration:3600 -b files:1 -f "host 192.168.111.1" -w d:\dumpedLog\day-month-year.pcap

2.1.6 Snort

Snort-IDS is the docker image extract from GNS3 repository. It is attached to R1-RFF to sniff the traffic / packets on the subnet 172.16.0.0. It is represented on topology list screenshot below:



Snort requires many configuration s to adapt to our project. The configuration of ICMP-Flood ,SYN-flood and port scan were already mentioned in my assignment 3. In our course project, I might want to add configuration about "web vulnerabilities":

2.1.7 Argus, TcpDump

Argus and Tcpdump can work with Wireshark together. Argus and Tcpdump installation and configuration were already menioned in my assignment 2 [3].

2.1.8 TCPReply and TCPRewrite

TCPReply and TCPRewirte can be used as post process of pcap datasets genreated from the first stage before Blue team enables all necessary security methods. Their installation and configuration were already mentioned in my assignment 3 [4].

2.1.9 UFW

https://www.vultr.com/docs/how-to-configure-ufw-firewall-on-ubuntu-14-04
on R1-RFF, I execute command "apt install ufw"

```
/ # apt install ufw
Reading package lists... Done
Building dependency tree
Reading state information... Done
ufw is already the newest version (0.36-1).
```

UFW configuration can be achieved by following command line, when I want to open SSH port only:

sudo ufw default deny
sudo ufw allow 22

2.1.2 ACL

On R2, ACL is the built in functionalitiy of Cisco Router. The configuration usages were already illustrated in my assignment 3 [4]. Here I only gave a sample:

access-list 101 deny udp any any eq snmptrap
access-list 101 permit tcp any any eq www

2.1.3 Iptables on RFF and Victim Machine

https://www.linode.com/docs/security/firewalls/control-network-traffic-with-iptables/
I set it as scanning all targeted devices every 1 minutes, so that the traffic can be consumed by wireshark used for question 2.

```
/ # apt-get install iptables
Reading package lists... Done
Building dependency tree
Reading state information... Done
iptables is already the newest version (1.8.2-4).
```

Iptables command is in /usr/sbin/ipables, and its usage can be represented as follow if I want to drop all the traffic coming from 192.168.111.1:

sudo iptables -A INPUT -s 192.168.111.1 -j DROP

2.1.4 Blockchain Based Blocklist (Tentatively)

Tentatively, I planned to apply Blockchain to record Blocklist and broadcast it to rest of network nodes. The idea is inspired by the reference paper [12].
The installation [7] on GNS3 topology nodes is complicated and companying with other issues. So, hopefully, I am able to make it work or partially work at the end of term.
The screenshot below shows Blockchain installed on one of nodes:

```
Deploying 'Blocklist'
---------------------
> transaction hash:    0x9a8b11d1e9e30ad78559ca9db42dd2227c618e5e6319d64234e82c5f17f9f8b3
> Blocks: 0            Seconds: 0
> contract address:    0xBd5fa0F904A944f8e1b7a08D1D2229297f5899a3
> block number:        2
> block timestamp:     1593978902
> account:             0xafCDb5c04700288706749B2AA9A3B2d1f76e4A1
> balance:             99.98564516
> gas used:            492505 (0x783d9)
> gas price:           20 gwei
> value sent:          0 ETH
> total cost:          0.0098501 ETH

> Saving migration to chain.
> Saving artifacts
-------------------------------------
> Total cost:          0.01435484 ETH
```

truffle(development)> instance = await Blocklist.at("0xBd5fa0F904A944f8e1b7a08D1D2229297f5899a3")

3 Penetration testing and defend sessions

3.1 Penetration testing planning:

I will follow the instruction shown below to conduct penetration testing:

Step 1: Startup Metasploit platform (msfconsole).

Step 2: Detect Victim Machine Opening ports:

msf5 > db_nmap -A 172.16.1.20

```
msf5 > db_nmap -A 172.16.1.20
[*] Nmap: Starting Nmap 7.70 ( https://nmap.org ) at 2020-07-15 21:05 +03
[*] Nmap: Note: Host seems down. If it is really up, but blocking our ping probe
s, try -Pn
[*] Nmap: Nmap done: 1 IP address (0 hosts up) scanned in 8.82 seconds
msf5 > db_nmap -A 172.16.1.20
[*] Nmap: Starting Nmap 7.70 ( https://nmap.org ) at 2020-07-15 21:11 +03
[*] Nmap: Nmap scan report for 172.16.1.20
```

We notice there are multiple ports opening.

Step 3: test 512, 513, 514 ports used by RSH services, I identified we can use "rlogin" command directly login victim machine without password.

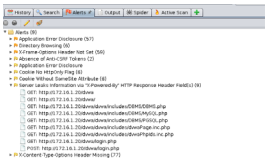
Step 4: Brutal-force the victim machine from lower-privilege account "tcptest".

Step 5: Conduct "Dirty Cow" script attack to escalate privilege to "root" user "bob".

Step 6: Metasploits output reports:

db_export -f xml /root/msfu/Exported.xml

Step 7: Identify "DVMN" web service vulnerabilities using ZAP. The results of "DVWA" web application vulnerabilities shown as below:



Step 8: attack "DVNM" web service through SQL injection, File inclusion, Brutal force etc.

Step 9: Load test on DVNM web service via "Hping3" ICMP and SYN flood command.

3.2 Defense Planning:

Step 1: Snort sniffling:

daemonlogger -i eth0 -i eth1 -o eth2

We execute this command to sniff traffic coming from R1-FRR at snort-IDS now:

snort -i eth0 -c /etc/snort/snort.conf -l /var/log/snort -A full

Step 2: Snort rules customization:

At this point, Snort is tested a lot from last section. Now, we need to write our own rules that will enable snort to detect more potential attack, such as DDoS, Port scan, brutal force, web vulnerability etc. Inspired by the instruction [10], I will configure my own rules.

Step 3: Netstat checking: To make it more precisely, I directly exectue the command "netstat" on vulerable targetted server "metasploitable2".

Step 4: ACL / Iptables configuration: we can specify ports and IP filters through ACL /Iptables configuration.

Step 5: UFW configuration: we can specify firewall policy via UFW installed on R1-RFF.

Step 6: Data collection via TCPReplay, wireshark, tcpdump, snort and Apache logs of web services

Step 7: Data Analytics via Argus, rgraph or Python notebook. Discover the most potential network issues sorted by severity. And we conclude the efficient and effective defend methods.

4 Evidence collection

Evidence Category	Collection method
PCAP traffic / packets	Wireshark; TCPDump; Tshark; TCPReplay
Snort Alerts / Dump logs	**daemonlogger -i eth0 -i eth1 -o eth2** **snort -i eth0 -c /etc/snort/snort.conf -l /var/log/snort -A full**
Web Services Logs	**Apache /var/apache2/logs/**

REFERENCES

[1] DR. NUR ZINCIR-HEYWOOD, https://web.cs.dal.ca/~zincir/cs6706.html
[2] Brightspace: https://dal.brightspace.com/d2l/home/124069
[3] https://linuxize.com/post/how-to-install-node-js-on-ubuntu-18.04/
[4] Install truffle suite: https://medium.com/@techgeek628/how-to-install-and-execute-truffle-on-an-ubuntu-16-04-7ebb3444707e
[5] Curl web services : https://www.baeldung.com/curl-rest
[6] Medusa guidelines: hackingarticles.in/comprehensive-guide-on-medusa-a-brute-forcing-tool/
[7] Truffle and Ganache-cli: https://medium.com/coinmonks/5-minute-guide-to-deploying-smart-contracts-with-truffle-and-ropsten-b3e30d5ee1e
[8] Snort default alerts: https://www.apt-browse.org/browse/debian/wheezy/main/all/snort-rules-default
[9] Metasplotable commands list: http://atic-tw.blogspot.com/p/userpass.html
[10] https://metasploit.help.rapid7.com/docs/metasploitable-2-exploitability-guide
[11] ZAP guild: https://www.zaproxy.org/docs/docker/about/
[12] Christian Gang Liu: Assignment 2: https://github.com/chrisliu01/network-management/blob/master/assignment%202%20-%20Christian%20Gang%20Liu%20-%20B00415613.pdf
[13] Christian Gang Liu: assignment 3: https://github.com/chrisliu01/network-management/blob/master/assignment%203%20-%20Christian%20Gang%20Liu%20-%20B00415613.pdf
[14] ZAP Concepts: https://en.wikipedia.org/wiki/OWASP_ZAP
[15] Dirty_cow source code : https://www.exploit-db.com/exploits/40839
[16] Privilege Esclastion: https://payatu.com/guide-linux-privilege-escalation
[17] Blockchain based DDoS attack mitigation: https://doi.org/10.1109/IBCAST.2019.8667147