Cybersecurity Blog

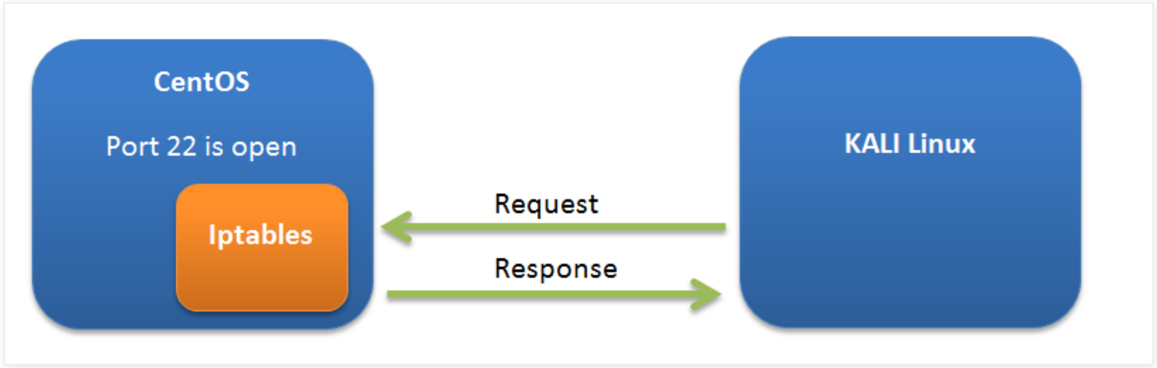
Everything about threat intelligence, blue team, red team, pentesting, security audit, security review, testing and assessment.

Thursday, May 1, 2014

Fool The Network Hunters (Hackers)

Portspoof is meant to be a lightweight, fast, portable and secure addition to the any firewall system or security system. The general goal of the program is to make the information gathering phase slow and bothersome for your attackers as much it is only possible. This is quite a change to the standard 5s Nmap scan, that will give a full view of your systems running services.

So let's start directly. So this is how the common structure of portspoof. First I will mention normal network structure without using portspoof and then with using portspoof. Below figure shows the normal structure of my network.



Here when attacker scans the CentOS server's network then first request will be sent to Iptables and according to iptables rule it will send the respond to the server. So currently in my iptables standard rules have been set.

Translate Language

Search

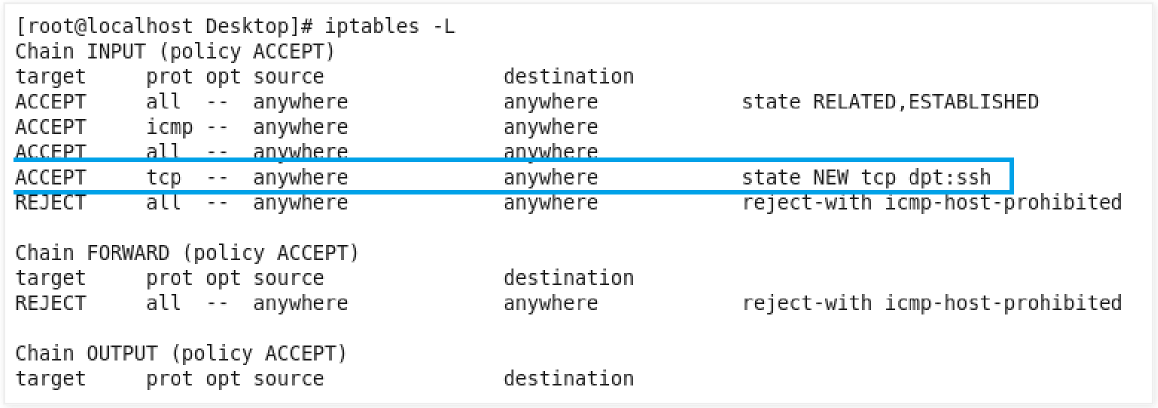
Search

Subscribe via email

Email address...
Submit

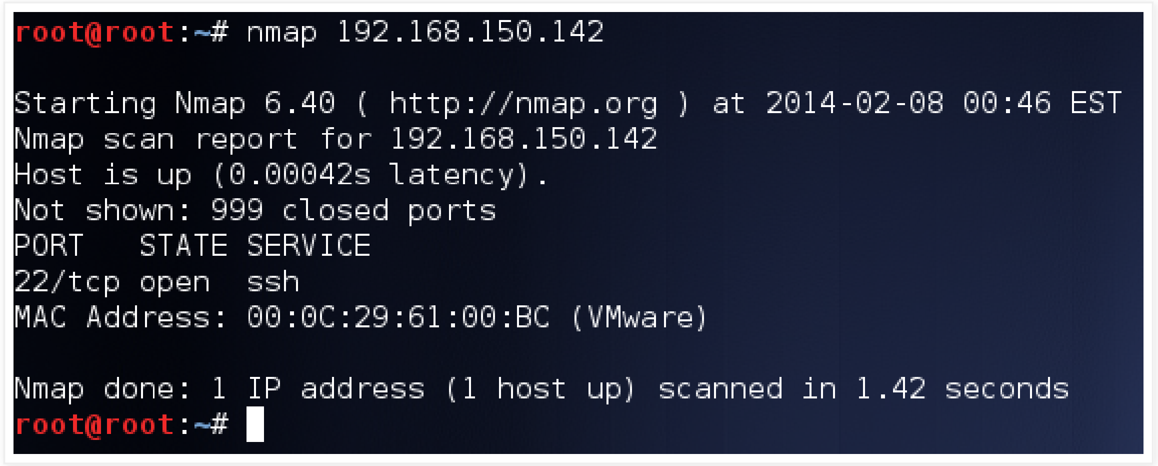
Blog Archive

- ► 2020 (2)
- ► 2019 (6)
- ► 2018 (4)
- ► 2017 (5)
- ► 2016 (11)
- ► 2015 (4)
- ▼ 2014 (22)
  - ► December 21 (1)
  - ► October 12 (1)
  - ► September 21 (3)
  - ► May 25 (1)
  - ► May 11 (1)
  - ▼ April 27 (1)
    - Fool The Network Hunters (Hackers)



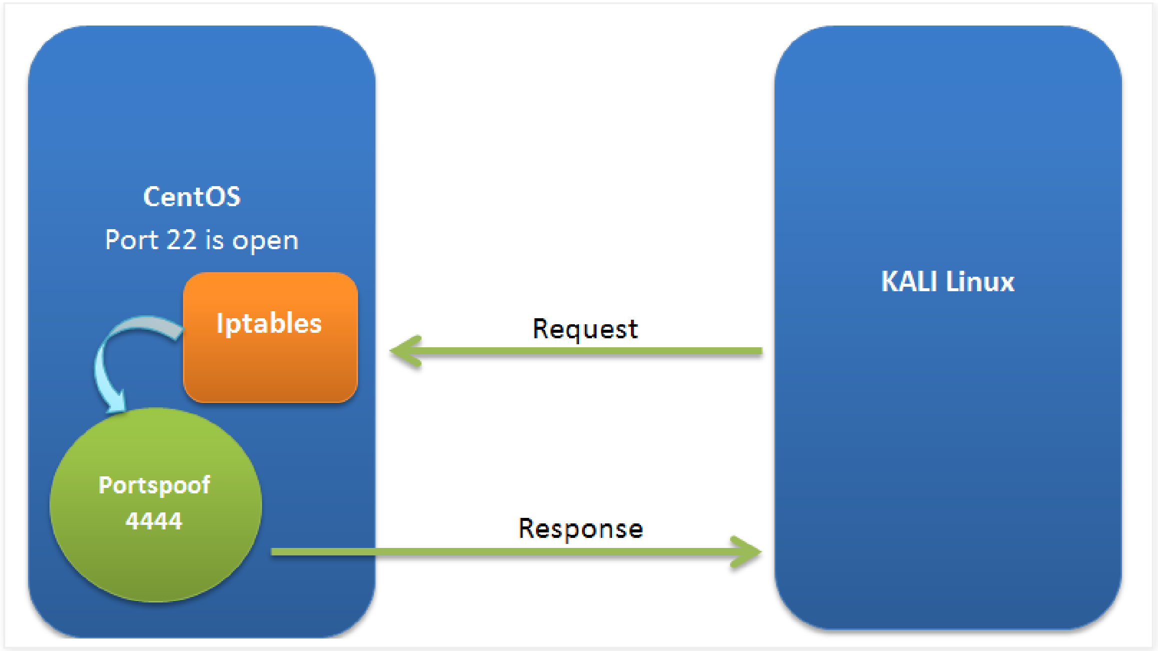
```
[root@localhost Desktop]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

As you can see port 22 is open and any connection thorough client machine to server's ssh service is allowed. So if I scan this network through NMAP from my KALI linux which I am considering as an attacker's machine then it will show me below result.



```
root@root:~# nmap 192.168.150.142

Starting Nmap 6.40 ( http://nmap.org ) at 2014-02-08 00:46 EST
Nmap scan report for 192.168.150.142
Host is up (0.00042s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:0C:29:61:00:BC (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.42 seconds
root@root:~#
```

Now let's check the network configuration with portspoof. Make sure portspoof's default port is 4444. You can change it according to your need.



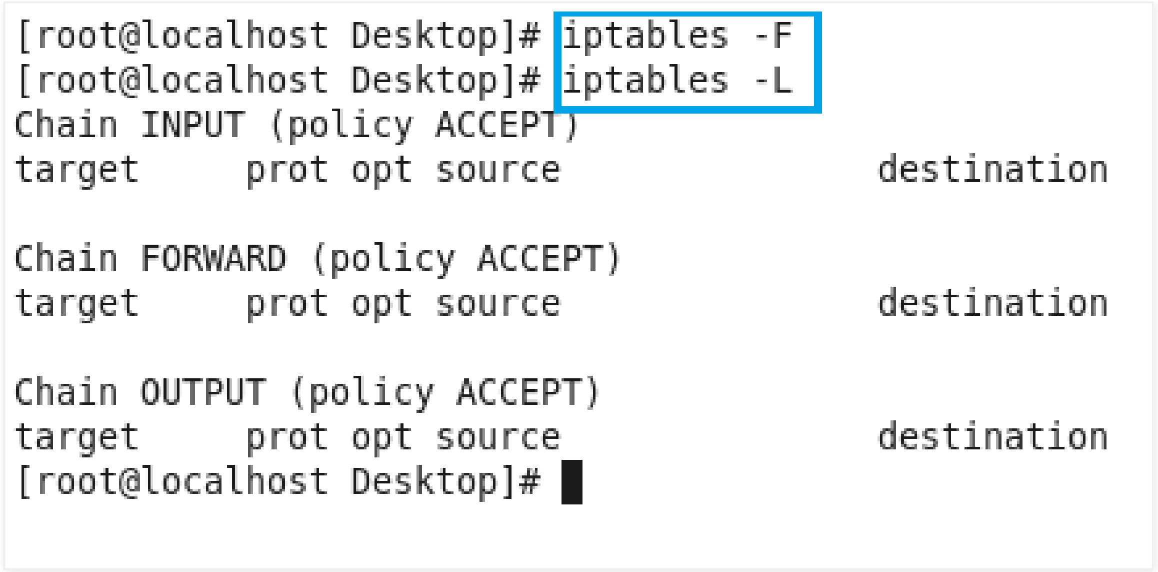
As we can see in the above picture when an attacker sends an nmap scan request to the centos server, then first it gets received to the iptables. Then rather than responding to the attacker machine, iptables forward that request to the portspoof on 4444 port and it enables portspoof to respond to the attacker's machine in order to show all 65535 ports open.

Let's Start Demonstration …
Firstly I am flushing all the rules of iptables and I am letting all packets allow into my network. To do so there is a following command shown below:

```
Iptables –f
```

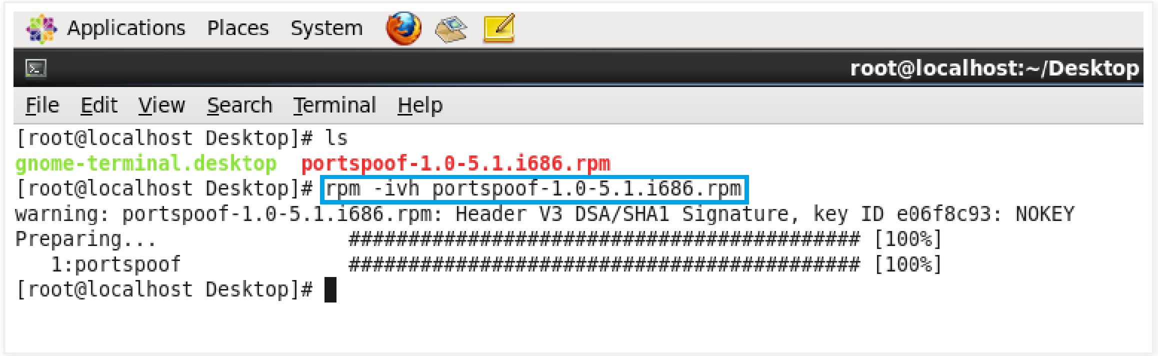
After giving this command if you want to see the current policy then you can check it with below command.

```
iptables –l
```


```
[root@localhost Desktop]# iptables -F
[root@localhost Desktop]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@localhost Desktop]# 
```

Now it's time to configure our iptables with the portspoof. For that lets download and install portspoof. I have downloaded rpm package of portspoof. Below command installs that package.

rpm –ivh portspoof-1.0-5.1.i686.rpm


Applications Places System
root@localhost:~/Desktop
File Edit View Search Terminal Help

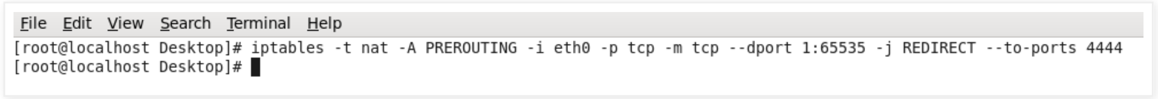
```
[root@localhost Desktop]# ls
gnome-terminal.desktop  portspoof-1.0-5.1.i686.rpm
[root@localhost Desktop]# rpm -ivh portspoof-1.0-5.1.i686.rpm
warning: portspoof-1.0-5.1.i686.rpm: Header V3 DSA/SHA1 Signature, key ID e06f8c93: NOKEY
Preparing...                ########################################### [100%]
   1:portspoof              ########################################### [100%]
[root@localhost Desktop]# 
```

Command/Command Option	Description
rpm	rpm package manager
-i	Install package
-v	Prints routine process verbose information
-h	Print 50 hash marks as the package archive is unpacked.

Next thing to do is to flush all the current firewall rules. As we already checked previously we do not have any rules set in our current firewall. So apply those iptables –f and –l command in your server to cross verify our process. Then as we can see from above pic that our firewall is up and running and it is allowing all packets from any network. Not it is time to forward those packets to portspoof in order to reply the client machine. To do so command is as follow.

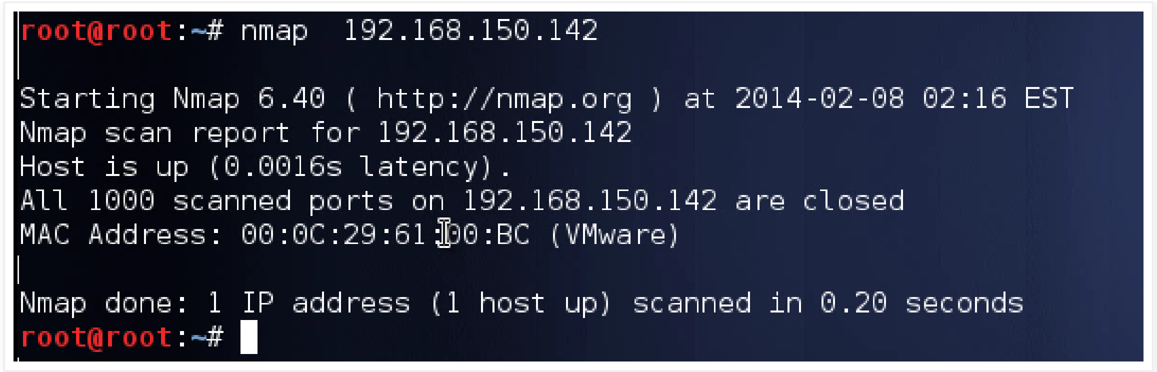
```
iptables –t nat  –A PREROUTING –i eth0 –p tcp –m tcp --dport 1:65535 –j REDIRECT --to-ports 4444
```

Command/Command Option	**Description**
Iptables	Linux default firewall.
-A	Appends the iptables rule to the end of the specified chain. This is the command used to add a rule when rule order in the chain does not matter.
-t	It specifies the table name which we are going to use.
-i	Selecting the interface.
-m	Additional match options are also available through modules loaded by the iptables command. To use a match option module, load the module by name using the -m option, such as -m (replacing with the name of the module).
-p	Sets the default policy for the specified chain, so that when packets traverse an entire chain without matching a rule, they are sent on to the specified target, such as ACCEPT or DROP.
--dport	Destination port
-j	Jump
--to-ports	Destination port to forward.



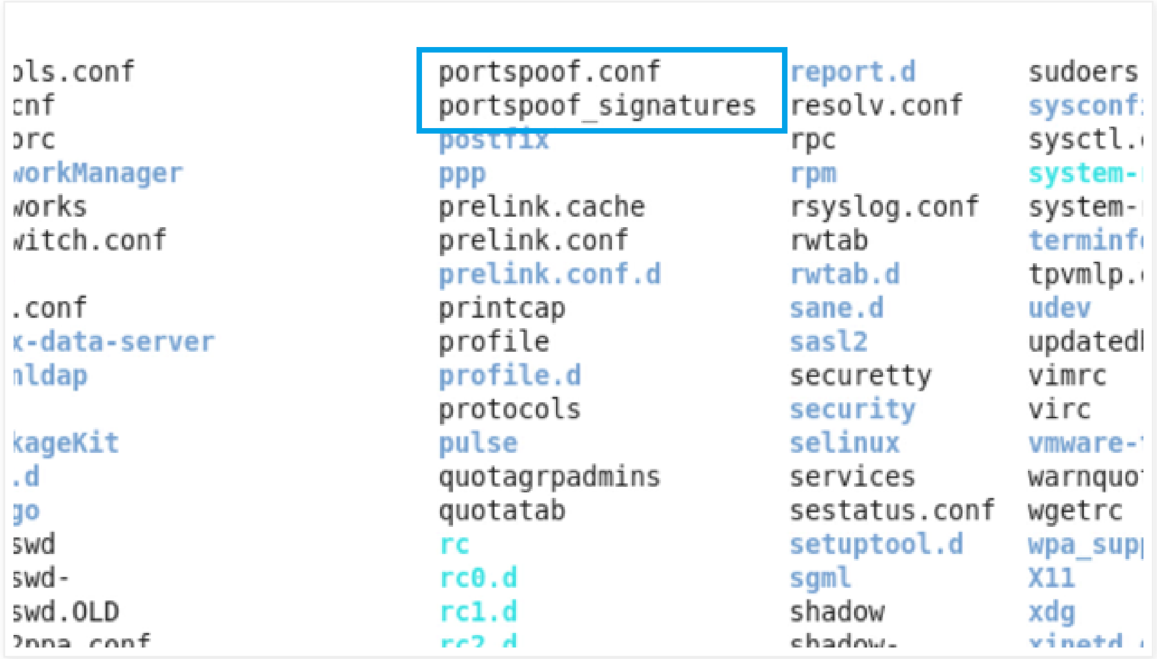
```
[root@localhost Desktop]# iptables -t nat -A PREROUTING -i eth0 -p tcp -m tcp --dport 1:65535 -j REDIRECT --to-ports 4444
[root@localhost Desktop]# 
```

Last few words in this command is very important that first it will collect all the packets accepted by an iptables and then it will forward it to the 4444 port which is a by default port of our portspoof tool.
Now let’s scan the target.(Keep in mind that we have not configured our portspoof.)



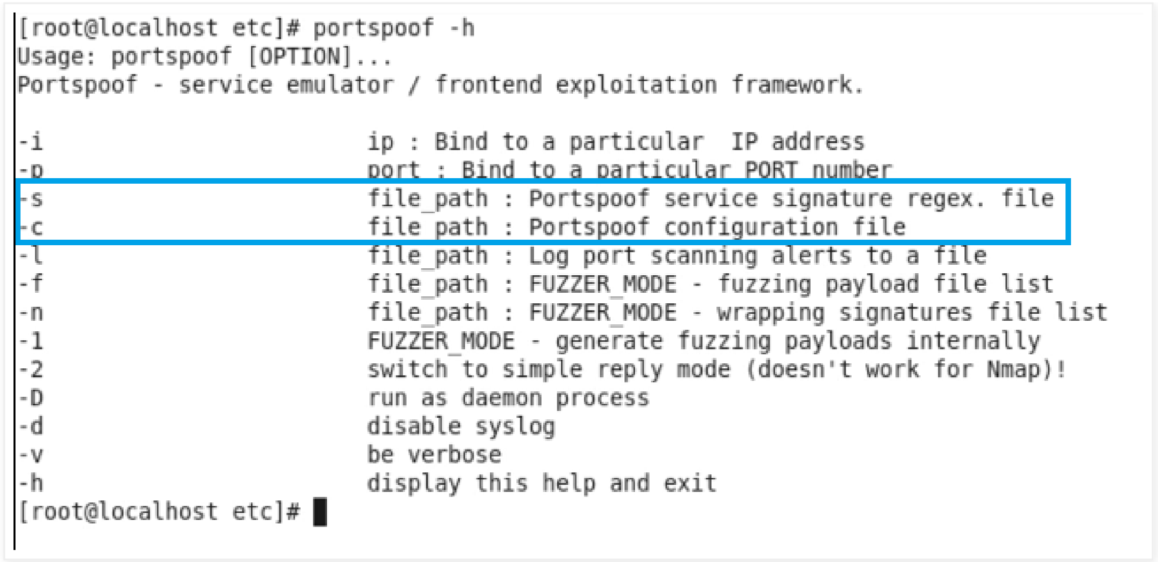
```
root@root:~# nmap  192.168.150.142

Starting Nmap 6.40 ( http://nmap.org ) at 2014-02-08 02:16 EST
Nmap scan report for 192.168.150.142
Host is up (0.0016s latency).
All 1000 scanned ports on 192.168.150.142 are closed
MAC Address: 00:0C:29:61:D0:BC (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.20 seconds
root@root:~#
```

Now it shows that our host(CentOS) is live and running but it is not showing any list of open port. It suggests that we have successfully configured our iptables with the portspoof. So our iptables is successfully sending all incoming packets to the portspoof. Now it is time to configure portspoof.
Portspoof runs with its two main files which is lying in /etc/ folder shown as below.



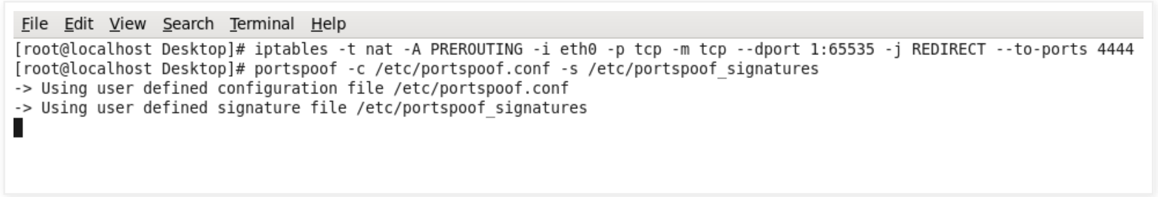
```
ols.conf          portspoof.conf          report.d       sudoers
cnf               portspoof_signatures    resolv.conf    sysconf
orc               postfix                 rpc            sysctl.
workManager       ppp                     rpm            system-
works             prelink.cache           rsyslog.conf   system-
witch.conf        prelink.conf            rwtab          terminf
                  prelink.conf.d          rwtab.d        tpvmlp.
.conf             printcap                sane.d         udev
x-data-server     profile                 sasl2          updatedl
nldap             profile.d               securetty      vimrc
                  protocols               security       virc
kageKit           pulse                   selinux        vmware-
.d                quotagrpadmins          services       warnquo
go                quotatab                sestatus.conf  wgetrc
swd               rc                      setuptool.d    wpa_sup
swd-              rc0.d                   sgml           X11
swd.OLD           rc1.d                   shadow         xdg
```

In config file, all the rules have been written that how and what should portspoof reply to the client machine and in signatures there are lots of signature of various scanning tools.
For example if I do normal nmap of 1 host, it will only show me port xyz is open and it will also show me port number. But, if I use nmap with –sV command then it will also try to fetch the service name which is used by the server and it will show me at the client side. So these –sV and like these other signatures are detected by the portspoof and it gives false results according to the request.
To start portspoof lets check the help to determine which options are provided to us.



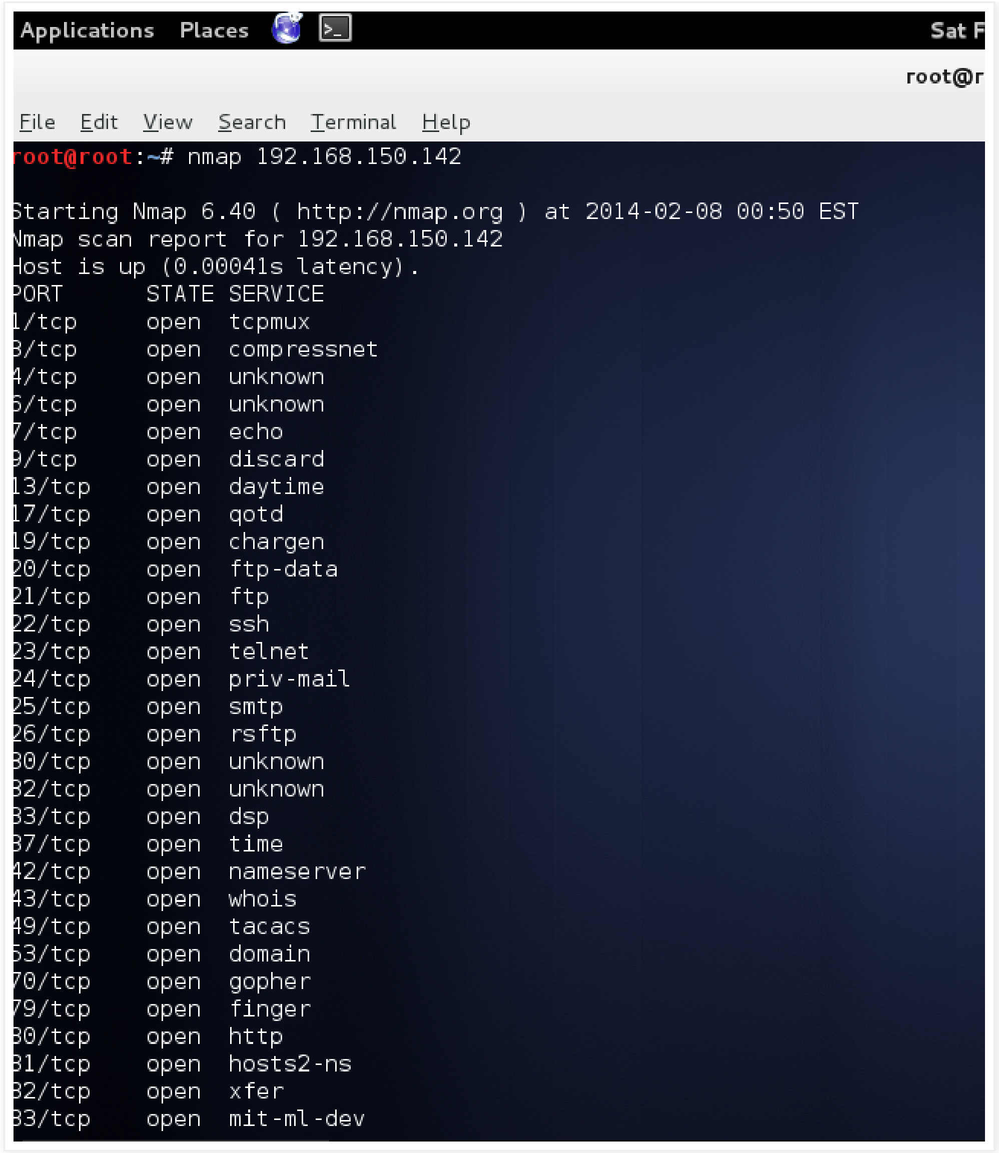
```
[root@localhost etc]# portspoof -h
Usage: portspoof [OPTION]...
Portspoof - service emulator / frontend exploitation framework.

-i                     ip : Bind to a particular  IP address
-p                     port : Bind to a particular PORT number
-s                     file_path : Portspoof service signature regex. file
-c                     file_path : Portspoof configuration file
-l                     file_path : Log port scanning alerts to a file
-f                     file_path : FUZZER_MODE - fuzzing payload file list
-n                     file_path : FUZZER_MODE - wrapping signatures file list
-1                     FUZZER_MODE - generate fuzzing payloads internally
-2                     switch to simple reply mode (doesn't work for Nmap)!
-D                     run as daemon process
-d                     disable syslog
-v                     be verbose
-h                     display this help and exit
[root@localhost etc]#
```

Two mandatory options are needed to run the portspoof. Command to run portspoof is as follow.
portspoof –c /etc/portspoof.conf –s /etc/portspoof_signatures
One you give this command it will look like as follow.



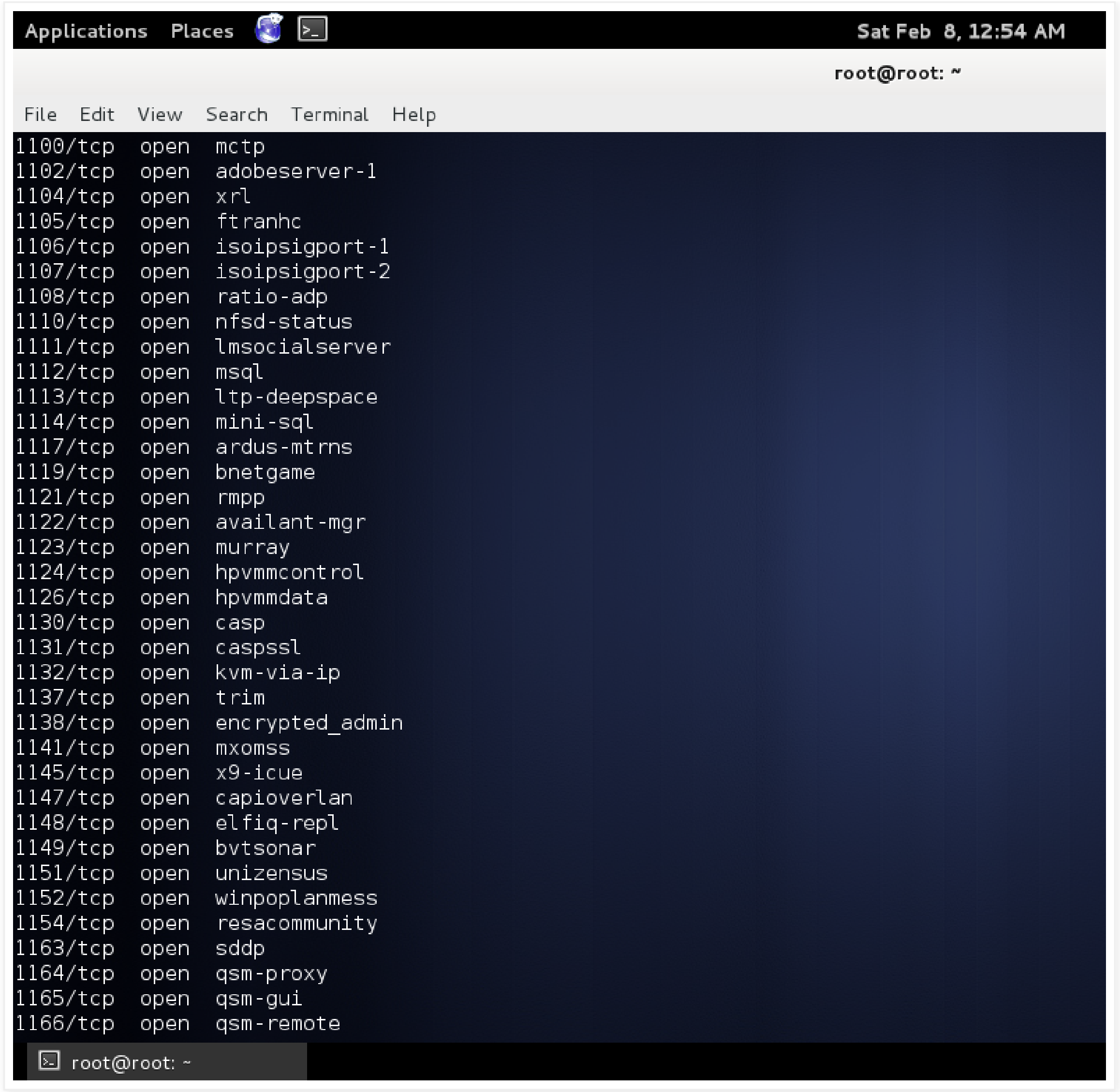
```
[root@localhost Desktop]# iptables -t nat -A PREROUTING -i eth0 -p tcp -m tcp --dport 1:65535 -j REDIRECT --to-ports 4444
[root@localhost Desktop]# portspoof -c /etc/portspoof.conf -s /etc/portspoof_signatures
-> Using user defined configuration file /etc/portspoof.conf
-> Using user defined signature file /etc/portspoof_signatures
```

Now it is time to scan from our attacker machine(Kali Linux).



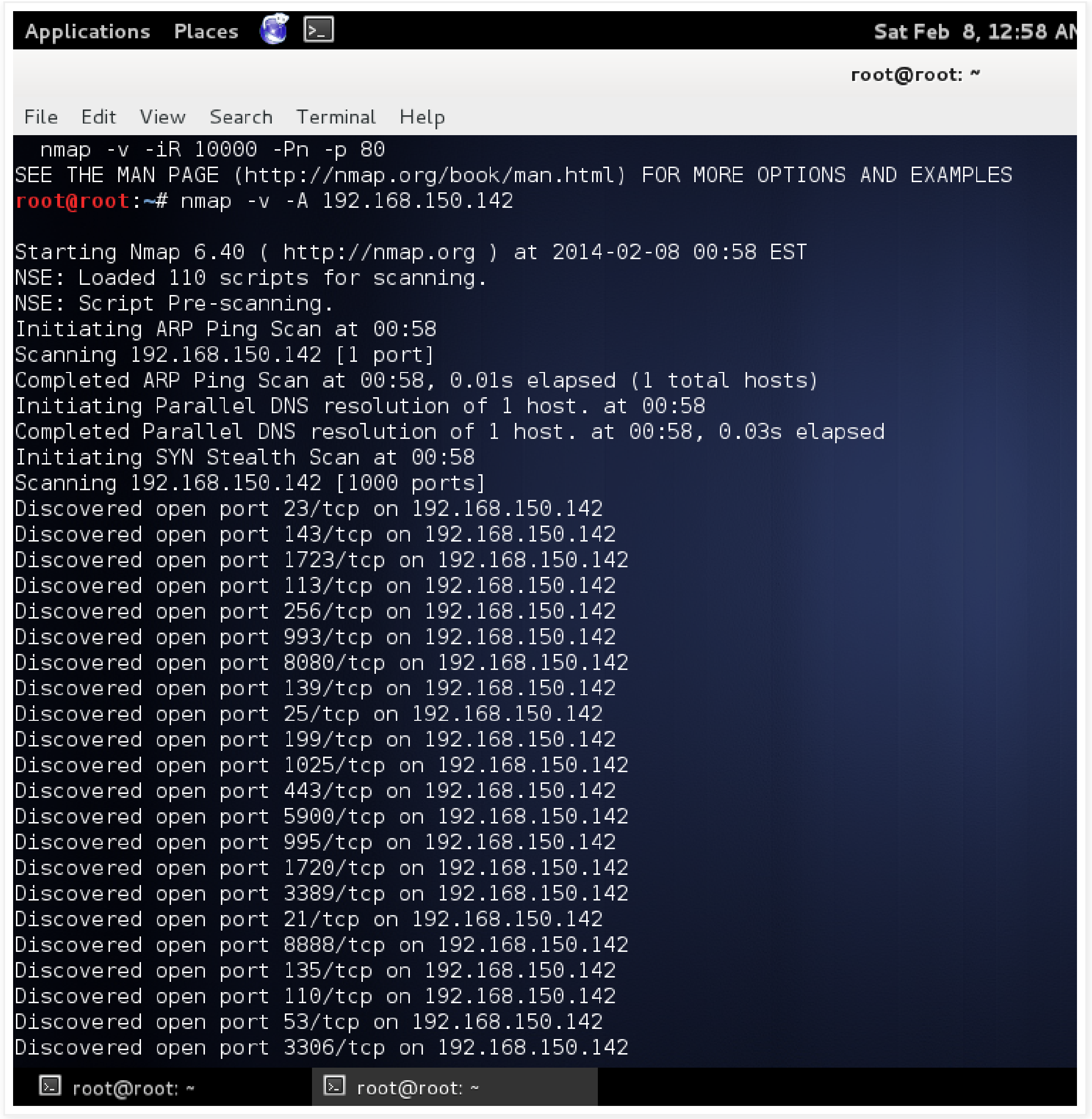
```
root@root:~# nmap 192.168.150.142

Starting Nmap 6.40 ( http://nmap.org ) at 2014-02-08 00:50 EST
Nmap scan report for 192.168.150.142
Host is up (0.00041s latency).
PORT      STATE SERVICE
1/tcp     open  tcpmux
3/tcp     open  compressnet
4/tcp     open  unknown
6/tcp     open  unknown
7/tcp     open  echo
9/tcp     open  discard
13/tcp    open  daytime
17/tcp    open  qotd
19/tcp    open  chargen
20/tcp    open  ftp-data
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
24/tcp    open  priv-mail
25/tcp    open  smtp
26/tcp    open  rsftp
30/tcp    open  unknown
32/tcp    open  unknown
33/tcp    open  dsp
37/tcp    open  time
42/tcp    open  nameserver
43/tcp    open  whois
49/tcp    open  tacacs
53/tcp    open  domain
70/tcp    open  gopher
79/tcp    open  finger
80/tcp    open  http
81/tcp    open  hosts2-ns
82/tcp    open  xfer
83/tcp    open  mit-ml-dev
```

As you can see starting from 1st, it will show all 65535 ports open. Actually these ports are not actually open or even exists but thus how we are fooling attacker to make him see all 65535 ports are opened.



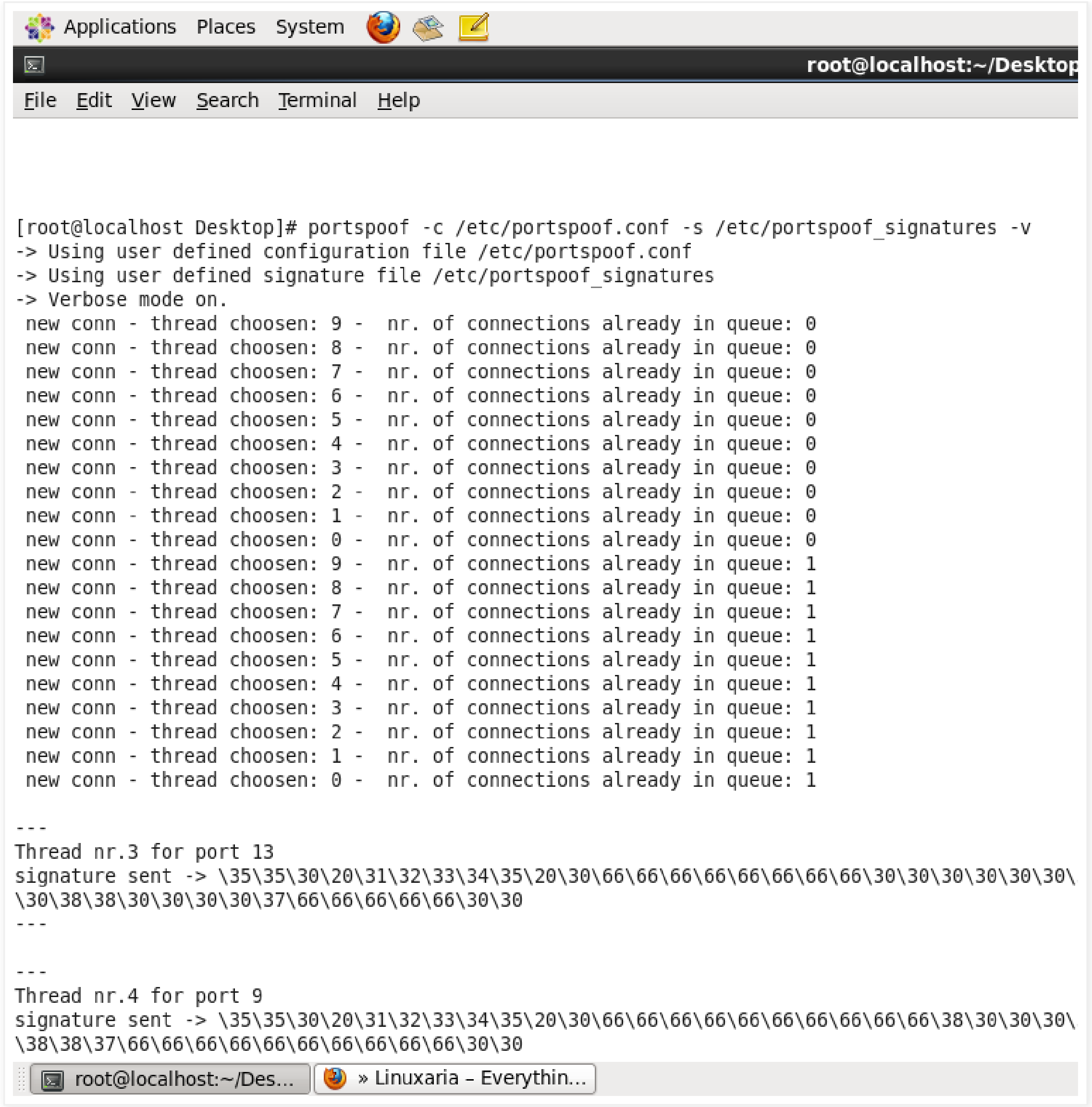
```
1100/tcp  open  mctp
1102/tcp  open  adobeserver-1
1104/tcp  open  xrl
1105/tcp  open  ftranhc
1106/tcp  open  isoipsigport-1
1107/tcp  open  isoipsigport-2
1108/tcp  open  ratio-adp
1110/tcp  open  nfsd-status
1111/tcp  open  lmsocialserver
1112/tcp  open  msql
1113/tcp  open  ltp-deepspace
1114/tcp  open  mini-sql
1117/tcp  open  ardus-mtrns
1119/tcp  open  bnetgame
1121/tcp  open  rmpp
1122/tcp  open  availant-mgr
1123/tcp  open  murray
1124/tcp  open  hpvmmcontrol
1126/tcp  open  hpvmmdata
1130/tcp  open  casp
1131/tcp  open  caspssl
1132/tcp  open  kvm-via-ip
1137/tcp  open  trim
1138/tcp  open  encrypted_admin
1141/tcp  open  mxomss
1145/tcp  open  x9-icue
1147/tcp  open  capioverlan
1148/tcp  open  elfiq-repl
1149/tcp  open  bvtsonar
1151/tcp  open  unizensus
1152/tcp  open  winpoplanmess
1154/tcp  open  resacommunity
1163/tcp  open  sddp
1164/tcp  open  qsm-proxy
1165/tcp  open  qsm-gui
1166/tcp  open  qsm-remote
```

If you want to scan that host with any signature within nmap then it will show as below. I am using nmap with –v and –A option. Then result will be as follows.



```
  nmap -v -iR 10000 -Pn -p 80
SEE THE MAN PAGE (http://nmap.org/book/man.html) FOR MORE OPTIONS AND EXAMPLES
root@root:~# nmap -v -A 192.168.150.142

Starting Nmap 6.40 ( http://nmap.org ) at 2014-02-08 00:58 EST
NSE: Loaded 110 scripts for scanning.
NSE: Script Pre-scanning.
Initiating ARP Ping Scan at 00:58
Scanning 192.168.150.142 [1 port]
Completed ARP Ping Scan at 00:58, 0.01s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 00:58
Completed Parallel DNS resolution of 1 host. at 00:58, 0.03s elapsed
Initiating SYN Stealth Scan at 00:58
Scanning 192.168.150.142 [1000 ports]
Discovered open port 23/tcp on 192.168.150.142
Discovered open port 143/tcp on 192.168.150.142
Discovered open port 1723/tcp on 192.168.150.142
Discovered open port 113/tcp on 192.168.150.142
Discovered open port 256/tcp on 192.168.150.142
Discovered open port 993/tcp on 192.168.150.142
Discovered open port 8080/tcp on 192.168.150.142
Discovered open port 139/tcp on 192.168.150.142
Discovered open port 25/tcp on 192.168.150.142
Discovered open port 199/tcp on 192.168.150.142
Discovered open port 1025/tcp on 192.168.150.142
Discovered open port 443/tcp on 192.168.150.142
Discovered open port 5900/tcp on 192.168.150.142
Discovered open port 995/tcp on 192.168.150.142
Discovered open port 1720/tcp on 192.168.150.142
Discovered open port 3389/tcp on 192.168.150.142
Discovered open port 21/tcp on 192.168.150.142
Discovered open port 8888/tcp on 192.168.150.142
Discovered open port 135/tcp on 192.168.150.142
Discovered open port 110/tcp on 192.168.150.142
Discovered open port 53/tcp on 192.168.150.142
Discovered open port 3306/tcp on 192.168.150.142
```

If you can remember when we started portspoof it was on verbose mode. So if we check that host machine now it will show some information related to log that which kinds of threads have been coming and which kind of signature reply that portspoof tool has given in respond to that request. This information will show as follow.


Applications Places System
root@localhost:~/Desktop
File Edit View Search Terminal Help
root@localhost:~/Des...
» Linuxaria – Everythin...

```
[root@localhost Desktop]# portspoof -c /etc/portspoof.conf -s /etc/portspoof_signatures -v
-> Using user defined configuration file /etc/portspoof.conf
-> Using user defined signature file /etc/portspoof_signatures
-> Verbose mode on.
 new conn - thread choosen: 9 -  nr. of connections already in queue: 0
 new conn - thread choosen: 8 -  nr. of connections already in queue: 0
 new conn - thread choosen: 7 -  nr. of connections already in queue: 0
 new conn - thread choosen: 6 -  nr. of connections already in queue: 0
 new conn - thread choosen: 5 -  nr. of connections already in queue: 0
 new conn - thread choosen: 4 -  nr. of connections already in queue: 0
 new conn - thread choosen: 3 -  nr. of connections already in queue: 0
 new conn - thread choosen: 2 -  nr. of connections already in queue: 0
 new conn - thread choosen: 1 -  nr. of connections already in queue: 0
 new conn - thread choosen: 0 -  nr. of connections already in queue: 0
 new conn - thread choosen: 9 -  nr. of connections already in queue: 1
 new conn - thread choosen: 8 -  nr. of connections already in queue: 1
 new conn - thread choosen: 7 -  nr. of connections already in queue: 1
 new conn - thread choosen: 6 -  nr. of connections already in queue: 1
 new conn - thread choosen: 5 -  nr. of connections already in queue: 1
 new conn - thread choosen: 4 -  nr. of connections already in queue: 1
 new conn - thread choosen: 3 -  nr. of connections already in queue: 1
 new conn - thread choosen: 2 -  nr. of connections already in queue: 1
 new conn - thread choosen: 1 -  nr. of connections already in queue: 1
 new conn - thread choosen: 0 -  nr. of connections already in queue: 1

---
Thread nr.3 for port 13
signature sent -> \35\35\30\20\31\32\33\34\35\20\30\66\66\66\66\66\66\66\66\30\30\30\30\30\
\30\38\38\30\30\30\30\37\66\66\66\66\66\30\30
---

---
Thread nr.4 for port 9
signature sent -> \35\35\30\20\31\32\33\34\35\20\30\66\66\66\66\66\66\66\66\38\30\30\30\
\38\38\37\66\66\66\66\66\66\66\66\66\66\30\30
```

Conclusion

Thus how you can fool the attacker or a noob. If you configure this he will be confused and out of his mind that which port is actually and legitimately open. If he is a pro noob then he will start hunting from port 1st to 65535(Hope so). So this is a very lightweight small tool to make attackers fool as well as to increase the amount of attack time with which you might trace the actual attacker.

References

1. http://linux.about.com/od/commands/l/blcmdl8_rpm.htm
   http://www.centos.org/docs/4/html/rhel-rg-en-4/s1-iptables-options.html

http://1.bp.blogspot.com/_UYgVWm4aYbE/R_GBndvsFvI/AAAAAAAAAW4/eEAY1j3SyIU/s1600-h/April-Fool-ILLUS.jpg

Posted by Frogy at 5/01/2014

Labels: april fool, nessus, nessus scan, network, network security, nmap, nmap scan, port scan

No comments:

Post a Comment

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)