

# NCS 490: Hardening Lab

By: Tony Perez

November 5, 2014

## Abstract

In this lab we learned and implemented how to secure our Linux machines. Security is important especially in today's world where break ins are common. Securing our systems require more of a setup but it will prevent unwanted visitors. It might not seem important and tedious until someone breaks in and steals or destroys our data. One of the most common ways is to implement and enforce better password policies. Other security measures are to not run things we do not need and to have an anti-virus. There are also network security measures we can implement to improve our system's security.

## 1 Introduction

For this lab we setup a password policy on both of our systems and made users use complex passwords. We needed to stop unnecessary services that were running in our system. We read about SELinux and how iptables protect our system. IPtables rules is our firewall and we needed to figure out what rules were currently set and also audit them using two tools called nmap and tcpdump.

## 2 System Hardening

Our password policy must have:

- Password length to more than 8 characters
- Password must have a mix of capital and lower case letters
- Password must have at least 1 number and 1 special character
- All passwords also expire after 30 days and the same password cannot be reused

To do this we had to edit two files as root. The first one was `/etc/login.defs`. Here we had to edit how long the password can be used for, the minimum number of days before a change, password character length and Number of days warning given before a password expires like so:

```
# Password aging controls:
#
#      PASS_MAX_DAYS   Maximum number of days a password may be used.
#      PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#      PASS_MIN_LEN     Minimum acceptable password length.
#      PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   30
PASS_MIN_DAYS   0
PASS_MIN_LEN    8
PASS_WARN_AGE   7
```

We set it so that the password expires after 30 days, they could change their password anytime, had to be at least 8 characters and will warn the user 7 days before his password would expire.

The second file we had to edit was the `/etc/pam.d/system-auth` file. This file is used implements the password complexity and how many times one can re-use their password.

```
##PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth      required      pam_env.so
auth      sufficient    pam_unix.so nullok try_first_pass
auth      requisite     pam_succeed_if.so uid >= 500 quiet
auth      required      pam_deny.so

account   required      pam_unix.so
account   sufficient    pam_localuser.so
account   sufficient    pam_succeed_if.so uid < 500 quiet
account   required      pam_permit.so

password  requisite     pam_cracklib.so try_first_pass retry=3 type= minlen=8 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1
password  sufficient    pam_unix.so sha512 shadow nullok try_first_pass use_authtok remember=1
password  required      pam_deny.so

session   optional     pam_keyinit.so revoke
session   required     pam_limits.so
session   [success=1 default=ignore] pam_succeed_if.so service in crond quiet use_uid
session   required     pam_unix.so
```

Here we edited the line:

```
# password requisite
```

And added these parameters:

```
# minlen=8 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1
```

**minlen=** forces the minimum amount of character

**ucredit=** forces the minimum amount letters

**lcredit=** forces the minimum amount case letters

**dcridit=** forces the minimum amount of numbers

**ocredit=** forces the minimum amount of symbols

We then tested it by trying to change our password with a password that did not follow our policy rule:

```
tony@pereztr-1 ~ $ passwd
Changing password for user tony.
Changing password for tony.
(current) UNIX password:
New password:
BAD PASSWORD: it is based on a dictionary word
New password:
BAD PASSWORD: it is based on a dictionary word
New password: 
```

Since our system was setup with a minimal version of CentOS we did not have to disable any services because the current services that were up were needed. To check we used the:

```
# chkconfig --list
```

### 3 Network Security

To improve our network security we first looked at our iptables rules and examined what rules we currently had. To see the current rules we used this command to list them:

```
# iptables -L
```

```
root@pereztr-1 ~ $ iptables -L
Chain INPUT (policy ACCEPT)
target    prot opt source                destination           state RELATED,ESTABLISHED
ACCEPT    all  --  anywhere              anywhere
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-with icmp-host-prohibited
REJECT    all  --  anywhere              anywhere

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

Here we see that the default rules are that it will **ACCEPT** packets on **all** ports from **anywhere** to **anywhere** that have been **ESTABLISHED** or **RELATED** which means that it will accept incoming packets that are part of an already established connection or related to and already established connection. Then we **ACCEPT** from the **icmp** port from **anywhere** to **anywhere**.

Next we go ahead and allow everything else in.

We also **ACCEPT** all **tcp** packets from and to **anywhere** that are from ssh, port 22.

The last thing in our **INPUT** chain is the we **REJECT** any network packet from any source with the icmp-host-prohibited message. This rule is used as the default DENY ALL rule at the end of any iptables chain to be sure that the only network packets that are allowed are those that are allowed by any iptables rules matched BEFORE this DENY ALL rule.(linuxcsv.com)

We will now add rules to our machines to allow logging.

```
# iptables -N LOGGING
```

This will create a chain called LOGGING

```
# iptables -I INPUT 7 -j LOGGING
```

We then add a rule to LOG into the INPUT chain

```
# iptables -A LOGGING -m limit --limit 10/min -j LOG --log-prefix "DROP: " --log-level 7
```

We then make it so we record 10 messages per minute and it will start with "DROP: " and will log anything (level 7).

```
# iptables -A LOGGING -j DROP
```

What ever we log we will then drop.

We save our current rules into `/etc/iptables.bak`

```
root@pereztr-1 ~ $ iptables-save > /etc/iptables.bak
root@pereztr-1 ~ $ cat /etc/iptables.bak
# Generated by iptables-save v1.4.7 on Tue Nov  4 17:48:33 2014
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [1723:340255]
:LOGGING - [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -p udp -m state --state NEW -m udp --dport 53 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 53 -j ACCEPT
-A INPUT -j LOGGING
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
-A LOGGING -m limit --limit 10/min -j LOG --log-prefix "DROP: " --log-level 7
-A LOGGING -j DROP
COMMIT
# Completed on Tue Nov  4 17:48:33 2014
root@pereztr-1 ~ $
```

Next we will flush our rules and show that we have no rules.

```
root@pereztr-1 ~ $ iptables -F
root@pereztr-1 ~ $ iptables -L -v
Chain INPUT (policy ACCEPT 32 packets, 4141 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain OUTPUT (policy ACCEPT 16 packets, 1632 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain LOGGING (0 references)
 pkts bytes target    prot opt in     out     source                   destination
root@pereztr-1 ~ $
```

Then we will restore our rules using our backup file.

```
root@pereztr-1 ~ $ iptables-restore < /etc/iptables.bak
root@pereztr-1 ~ $ iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in     out    source            destination        state
 6  456 ACCEPT    all  --  any    any    anywhere          anywhere           state RELATED,ESTABLISHED
 0    0 ACCEPT    icmp --  any    any    anywhere          anywhere
 0    0 ACCEPT    all  --  lo     any    anywhere          anywhere
 0    0 ACCEPT    tcp  --  any    any    anywhere          anywhere           state NEW tcp dpt:ssh
 0    0 ACCEPT    udp  --  any    any    anywhere          anywhere           state NEW udp dpt:domain
 0    0 ACCEPT    tcp  --  any    any    anywhere          anywhere           state NEW tcp dpt:domain
 0    0 LOGGING   all  --  any    any    anywhere          anywhere
 0    0 REJECT    all  --  any    any    anywhere          anywhere           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in     out    source            destination        state
 0    0 REJECT    all  --  any    any    anywhere          anywhere           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT 4 packets, 544 bytes)
pkts bytes target      prot opt in     out    source            destination

Chain LOGGING (1 references)
pkts bytes target      prot opt in     out    source            destination        limit
 0    0 LOG        all  --  any    any    anywhere          anywhere           limit: avg 10/min burst 5 LOG level debug prefix `DROP: '
 0    0 DROP       all  --  any    any    anywhere          anywhere
root@pereztr-1 ~ $
```

Now we will use the nmap tool on our first machine, 10.103.67.80, from our second one, 10.103.67.81.

<pre>root@pereztr-1 ~ \$ nmap 10.103.67.81  Starting Nmap 5.51 ( http://nmap.org ) at 2014-11-04 18:17 EST Nmap scan report for ncs490ln-pereztr-2.cs.sunyit.edu (10.103.67.81) Host is up (0.0031s latency). Not shown: 998 filtered ports PORT      STATE SERVICE 22/tcp    open  ssh 53/tcp    open  domain MAC Address: 4A:87:91:4E:D5:A1 (Unknown)  Nmap done: 1 IP address (1 host up) scanned in 5.12 seconds</pre>	<pre>root@pereztr-1 ~ \$ nmap 10.103.67.81  Starting Nmap 5.51 ( http://nmap.org ) at 2014-11-04 18:17 EST Nmap scan report for ncs490ln-pereztr-2.cs.sunyit.edu (10.103.67.81) Host is up (0.0020s latency). Not shown: 998 closed ports PORT      STATE SERVICE 22/tcp    open  ssh 53/tcp    open  domain MAC Address: 4A:87:91:4E:D5:A1 (Unknown)  Nmap done: 1 IP address (1 host up) scanned in 1.25 seconds</pre>
--	--

(a) iptables enabled

(b) iptables disabled

The difference we see here is that when the when iptables are disabled they are marked as closed in the not shown section. When iptables is enabled they are marked as filtered.

The ports that are open are ssh(22) and dns(53).

To get more information using nmap we use the **-A** flag.

```
root@pereztr-2 ~ $ nmap -A 10.103.67.80

Starting Nmap 5.51 ( http://nmap.org ) at 2014-11-04 18:13 EST
Nmap scan report for ncs490ln-pereztr-1.cs.sunyit.edu (10.103.67.80)
Host is up (0.00075s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.3 (protocol 2.0)
| ssh-hostkey: 1024 cc:4b:d9:b1:5d:4d:ed:8d:ff:d0:1f:9a:2d:c5:20:7f (DSA)
|_ 2048 38:4b:d1:11:3e:c2:2c:86:41:51:ef:10:ff:dc:cf:70 (RSA)
53/tcp    open  domain
MAC Address: C2:EA:63:67:CB:E3 (Unknown)
No exact OS matches for host (If you know what OS is running on it, see http://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=5.51%0=11/4%0T=22%CT=1%CU=37994%PV=Y%DS=1%DC=D%G=Y%M=C2EA63%TM=54
OS:595DB%CP=x86_64-redhat-linux-gnu)SEQ(SP=107%GCD=1%ISR=107%TI=Z%CI=I
OS:%TS=A)OPS(O1=MSB4ST11NW5%02=MSB4ST11NW5%03=MSB4NNT11NW5%04=MSB4ST11NW5%0
OS:5=MSB4ST11NW5%06=MSB4ST11)WIN(W1=3890%W2=3890%W3=3890%W4=3890%W5=3890%W6
OS:=3890)ECN(R=Y%DF=Y%T=40%W=3908%0=MSB4NNSNW5%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0
OS:%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%0=%RD=
OS:0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%0=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%
OS:S=A%A=Z%F=R%0=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%0=%RD=0%Q=)U1(
OS:R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=
OS:N%T=40%CD=S)

Network Distance: 1 hop

TRACEROUTE
HOP RTT ADDRESS
1 0.75 ms ncs490ln-pereztr-1.cs.sunyit.edu (10.103.67.80)

OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.69 seconds
```

Figure 2: First Machine

```
root@pereztr-1 ~ $ nmap -A 10.103.67.81

Starting Nmap 5.51 ( http://nmap.org ) at 2014-11-04 18:22 EST
Nmap scan report for ncs490ln-pereztr-2.cs.sunyit.edu (10.103.67.81)
Host is up (0.00091s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.3 (protocol 2.0)
| ssh-hostkey: 1024 cc:4b:d9:b1:5d:4d:ed:8d:ff:d0:1f:9a:2d:c5:20:7f (DSA)
|_ 2048 38:4b:d1:11:3e:c2:2c:86:41:51:ef:10:ff:dc:cf:70 (RSA)
53/tcp    open  domain
MAC Address: 4A:87:91:4E:D5:A1 (Unknown)
No exact OS matches for host (If you know what OS is running on it, see http://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=5.51%0=11/4%0T=22%CT=1%CU=30877%PV=Y%DS=1%DC=D%G=Y%M=4A8791%TM=54
OS:595FB8%CP=x86_64-redhat-linux-gnu)SEQ(SP=107%GCD=1%ISR=108%TI=Z%CI=I
OS:%TS=A)OPS(O1=MSB4ST11NW5%02=MSB4ST11NW5%03=MSB4NNT11NW5%04=MSB4ST11NW5%0
OS:5=MSB4ST11NW5%06=MSB4ST11)WIN(W1=3890%W2=3890%W3=3890%W4=3890%W5=3890%W6
OS:=3890)ECN(R=Y%DF=Y%T=40%W=3908%0=MSB4NNSNW5%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0
OS:%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%0=%RD=
OS:0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%0=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%
OS:S=A%A=Z%F=R%0=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%0=%RD=0%Q=)U1(
OS:R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=
OS:N%T=40%CD=S)

Network Distance: 1 hop

TRACEROUTE
HOP RTT ADDRESS
1 0.91 ms ncs490ln-pereztr-2.cs.sunyit.edu (10.103.67.81)

OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 27.73 seconds
```

Figure 3: Second Machine

We will now use the tcpdump tool to sniff network traffic.

To do this we will run the tcpdump command, save the output into a file which we called tcpdumpOP.pcap, and then used the tcpdump command to read it. When running the tcpdump command we will have to make it run in the background so we can use links to search the web on our Linux machine and produce some traffic.

```
root@pereztr-1 ~ $ tcpdump -w tcpdumpOP.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
^Z
[1]+  Stopped                  tcpdump -w tcpdumpOP.pcap
root@pereztr-1 ~ $ bg 1
[1]+  tcpdump -w tcpdumpOP.pcap &
root@pereztr-1 ~ $ jobs
[1]+  Running                  tcpdump -w tcpdumpOP.pcap &
```

Here we started the tcpdump with flag -w to write it to a file used **CTRL-Z** to stop it then **bg 1** to the background. We checked to see if it was running with the **jobs** command.

We now use links to search the web.

```
# links
```

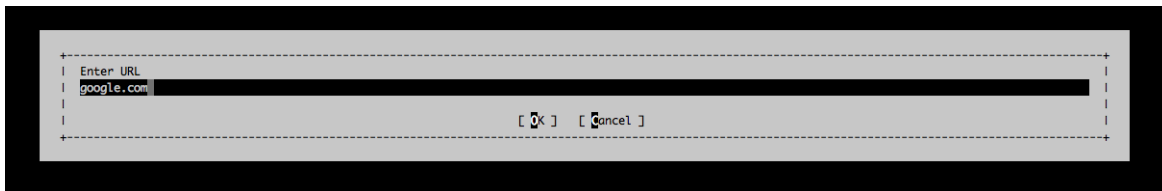


Figure 4: Going to google.com

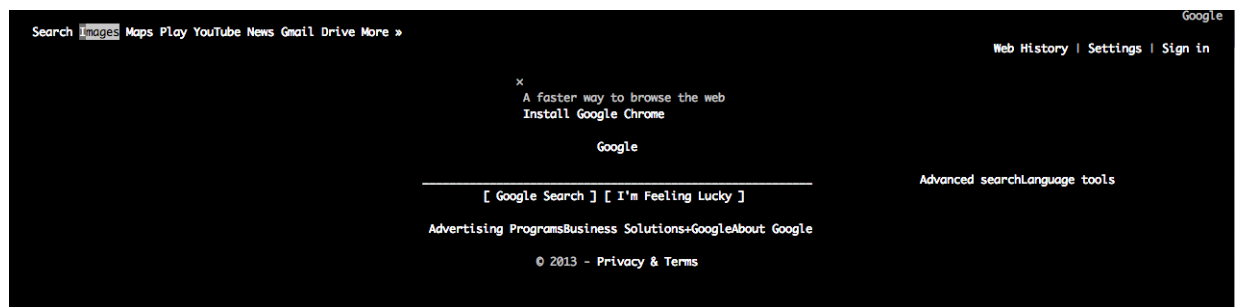


Figure 5: searching Linux



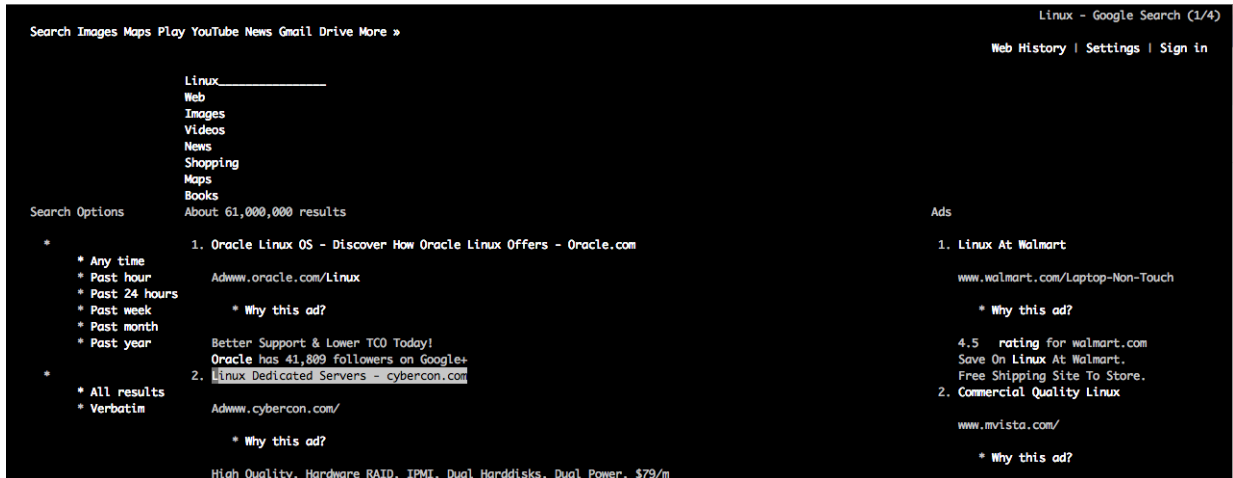


Figure 6: Selecting cybercron.com website

Now we will exit links and stop the tcpdump process. By bringing the tcpdump to the foreground then using **CTRL-C** to kill it.

```
# fg 1
```

To display the output of the tcpdump that we saved use this command:

```
# tcpdump -r tcpdump0P.pcap
```

```
12:05:39.458070 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [..], ack 928, win 1040, options [nop,nop,TS val 699913623 ecr 159145154], length 0
12:05:39.645159 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 00:0f:d3:05:27:e6 (oui Unknown), length 548
12:05:40.174816 ARP, Request who-has 10.107.2.130 tell 10.107.2.128, length 28
12:05:40.195501 ARP, Request who-has 10.103.0.32 tell tel500-kondapd-web.cs.sunyit.edu, length 28
12:05:40.412442 STP 802.1d, Config, Flags [none], bridge-id 8323.44:03:a7:59:b5:80.800c, length 42
12:05:41.067019 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [P..], seq 529:577, ack 928, win 1040, options [nop,nop,TS val 699915232 ecr 159145154], length 48
12:05:41.068921 IP ncs490ln-pereztr-1.cs.sunyit.edu.ssh > fang.cs.sunyit.edu.38190: Flags [P..], seq 928:976, ack 577, win 1086, options [nop,nop,TS val 159146864 ecr 699915232], length 48
12:05:41.167736 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [..], ack 976, win 1040, options [nop,nop,TS val 699915333 ecr 159146864], length 0
12:05:41.195444 ARP, Request who-has 10.103.0.32 tell tel500-kondapd-web.cs.sunyit.edu, length 28
12:05:41.356026 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 00:19:b9:cd:9d:28 (oui Unknown), length 300
12:05:41.598848 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [P..], seq 577:625, ack 976, win 1040, options [nop,nop,TS val 699915763 ecr 159146864], length 48
12:05:41.608917 IP ncs490ln-pereztr-1.cs.sunyit.edu.ssh > fang.cs.sunyit.edu.38190: Flags [P..], seq 976:1024, ack 625, win 1086, options [nop,nop,TS val 159147396 ecr 699915763], length 48
12:05:41.707701 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [..], ack 1024, win 1040, options [nop,nop,TS val 699915873 ecr 159147396], length 0
12:05:41.962410 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [P..], seq 625:673, ack 1024, win 1040, options [nop,nop,TS val 699916126 ecr 159147396], length 48
12:05:41.963891 IP ncs490ln-pereztr-1.cs.sunyit.edu.ssh > fang.cs.sunyit.edu.38190: Flags [P..], seq 1024:1104, ack 673, win 1086, options [nop,nop,TS val 159147759 ecr 699916126], length 80
12:05:42.067391 IP fang.cs.sunyit.edu.38190 > ncs490ln-pereztr-1.cs.sunyit.edu.ssh: Flags [..], ack 1104, win 1040, options [nop,nop,TS val 699916232 ecr 159147759], length 0
12:05:42.195765 ARP, Request who-has 10.103.0.32 tell tel500-kondapd-web.cs.sunyit.edu, length 28
12:05:42.443076 STP 802.1d, Config, Flags [none], bridge-id 8323.44:03:a7:59:b5:80.800c, length 42
```

Figure 7: tcpdump output

Here we can see ARP, STP, and IP packets that were captured. The ARP shows its "Who has" message and where it is going to. We can also see the length of the packet, flags, sequence number and time.

Now we will read the tcpdump output file but show http traffic only.

```
# tcpdump -r tcpdump0P.pcap port http
```

```
12:06:15.657064 IP ncs490ln-pereztr-1.cs.sunyit.edu.32944 > www.cybercon.com.http: Flags [..], ack 45145, win 2003, options [nop,nop,TS val 159181452 ecr 3558692752], length 0
12:06:15.657072 IP www.cybercon.com.http > ncs490ln-pereztr-1.cs.sunyit.edu.32944: Flags [..], seq 45145:46513, ack 343, win 122, options [nop,nop,TS val 3558692778 ecr 159181395], length 1368
12:06:15.693112 IP www.cybercon.com.http > ncs490ln-pereztr-1.cs.sunyit.edu.32944: Flags [..], seq 46513:47881, ack 343, win 122, options [nop,nop,TS val 3558692810 ecr 159181427], length 1368
12:06:15.693127 IP ncs490ln-pereztr-1.cs.sunyit.edu.32944 > www.cybercon.com.http: Flags [..], ack 47881, win 2003, options [nop,nop,TS val 159181488 ecr 3558692778], length 0
12:06:15.693135 IP www.cybercon.com.http > ncs490ln-pereztr-1.cs.sunyit.edu.32944: Flags [..], seq 47881:49249, ack 343, win 122, options [nop,nop,TS val 3558692810 ecr 159181427], length 1368
12:06:15.693140 IP www.cybercon.com.http > ncs490ln-pereztr-1.cs.sunyit.edu.32944: Flags [P..], seq 49249:49614, ack 343, win 122, options [nop,nop,TS val 3558692810 ecr 159181427], length 365
12:06:15.693198 IP ncs490ln-pereztr-1.cs.sunyit.edu.32944 > www.cybercon.com.http: Flags [..], ack 49614, win 2003, options [nop,nop,TS val 159181488 ecr 3558692810], length 0
12:06:16.527436 IP www.cybercon.com.http > ncs490ln-pereztr-1.cs.sunyit.edu.32945: Flags [F..], seq 6293, ack 304, win 122, options [nop,nop,TS val 3558693646 ecr 159181319], length 0
12:06:16.552433 IP www.cybercon.com.http > ncs490ln-pereztr-1.cs.sunyit.edu.32944: Flags [F..], seq 49614, ack 343, win 122, options [nop,nop,TS val 3558693669 ecr 159181488], length 0
12:06:16.566834 IP ncs490ln-pereztr-1.cs.sunyit.edu.32945 > www.cybercon.com.http: Flags [..], ack 6294, win 904, options [nop,nop,TS val 159182362 ecr 3558693646], length 0
12:06:16.595798 IP ncs490ln-pereztr-1.cs.sunyit.edu.32944 > www.cybercon.com.http: Flags [..], ack 49615, win 2003, options [nop,nop,TS val 159182391 ecr 3558693669], length 0
12:06:19.191357 IP ncs490ln-pereztr-1.cs.sunyit.edu.32944 > www.cybercon.com.http: Flags [F..], seq 343, ack 49615, win 2003, options [nop,nop,TS val 159184986 ecr 3558693669], length 0
12:06:19.191482 IP ncs490ln-pereztr-1.cs.sunyit.edu.32945 > www.cybercon.com.http: Flags [F..], seq 304, ack 6294, win 904, options [nop,nop,TS val 159184986 ecr 3558693646], length 0
12:06:19.191554 IP ncs490ln-pereztr-1.cs.sunyit.edu.44381 > ord08s13-in-f18.1e100.net.http: Flags [F..], seq 1660, ack 73471, win 2003, options [nop,nop,TS val 159184986 ecr 2525858931], length 0
12:06:19.191594 IP ncs490ln-pereztr-1.cs.sunyit.edu.46477 > ord08s13-in-f5.1e100.net.http: Flags [F..], seq 394, ack 577, win 493, options [nop,nop,TS val 159184986 ecr 2525844463], length 0
12:06:19.229575 IP ord08s13-in-f18.1e100.net.http > ncs490ln-pereztr-1.cs.sunyit.edu.44381: Flags [F..], seq 73471, ack 1661, win 361, options [nop,nop,TS val 2525869132 ecr 159184986], length 0
```

Figure 8: Here we can see the packets we sent and received from cybercron.com

## 4 Conclusion

This lab has shown us how to improve our systems security by implementing the use of better password, killing unneeded process, and using iptables to allow only what we want. As we go on we will have to start allowing other service to our iptables and so this gave us a good understanding of how they work. We then used nmap and tcpdump to see what is going on with our machines.