ISP/NSP Security Workshop

APRICOT 2007

Bali, Indonesia

21 February – 2 March 2007

Part 1 : Host Security

Before we begin

If you are running the X window as root

First create a normal user

```
# adduser
```

Or

```
# pw user add username -m
# passwd username
# chmod 700 /home/username
```

Logout and re login as a normal user

su access

```
% su -
su: Sorry
```

Now from root console do

```
# pw user mod username -G wheel
```

Now open a new virtual terminal using normal account
Verify you belong to wheel group

```
% id
% su -
```

You should now be able to su to root
Use root access only when required. All other tasks must be executed as a normal user account.

Note:

- % sign means execute command as Normal user
- # Sign means execute command as Super user
- `% whoami`
  - means this is a command that needs to be executed as normal user
- `% id` *`username`*
  - here *`username`*(italics) must be replaced with your input
- If you are not comfortable using the vi editor please install pico
  - `#pkg_add ftp://`*`192.168.0.1`*`/pub/FreeBSD/ports/i386/packages-6.2-release/Latest/pico.tbz`

Configure TcpWrapper

Allow ssh from local network and deny the rest

```
# vi /etc/hosts.allow
```

```
ALL : 127.0.0.1 : allow
ALL : [::1] : allow
sshd: 192.168.0. : allow
ALL : ALL : deny
```

Check if sshd is running

```
# ps ax | grep ssh
```

If SSH is not running, run it

```
# vi /etc/rc.conf
```

```
sshd_enable=”YES”
```

```
# /etc/rc.d/sshd start
```

SSH

Remote login

```
%ssh username@remote_ssh_server
```

X forwarding

```
% xhost + local:
%ssh -X username@remote_ssh_server
% env | grep DISPLAY
% xlogo
```

Compression

```
ssh -C username@remote_ssh_server
```

TCP Forwarding
-N Tunnel only mode
-f Run in background

```
ssh -N -f -L local-port:remote-host:remote-port user@remote-host
```

Example:

```
% ssh -N -f -L 8080:192.168.0.1:80 user@192.168.0.2
```

Secure Copy

```
%scp filename.txt server:
```

Secure FTP

```
%sftp server_name
```

Public key based SSH Authentication
In Client:

```
% ssh-keygen
```

Make sure you put a pass phrase

```
% cd ~/.ssh
% scp id_rsa.pub server:.ssh/
```

In server:

```
% cd ~/.ssh
% cat id_rsa.pub >> authorized_keys
```

In client:

```
% ssh username@server
```

Enter the pass phrase for the private key instead of the password of the remote server

Syslog

Enable NTP

```
# vi /etc/rc.conf
```

```
ntpdate_enable=YES
ntpdate_flag="-b 192.168.0.1"
```

Remote logging
Add the following line to /etc/syslog.conf
Use the peer computer as the remote host and test sending syslog output to each other

```
# vi /etc/syslog.conf
```

```
*.*     @remote_host
```

Restart syslog client

```
# killall -HUP syslogd
```

Verify that you are getting messages from syslog clients who has configured you as a server. For clients sending logs to another server please check the log files in the server

```
# tail -f /var/log/messages
```

Note: Make sure that on the Syslog server the syslogd daemon is running without "-ss" option

Server Monitoring

check the uptime,load average
```
%uptime
```

check the process running in the server
```
%ps -aux
```

check the process running in the server which updates information near real time
```
%top
```

Using netstat to look at all sockets
```
% netstat -an
```

Using netstat to look at current tcp sockets
```
% netstat -n -p tcp
```

using sockstat to see Ipv4 listening sockets(only on BSD)
```
# sockstat -4 -l
```

Process Accounting

Enable Process Accounting
```
#vi /etc/rc.conf
```
```
accounting_enable="YES"
```

Per process stats
```
$ sa
```

Per user stats
```
$ sa -m
```

Per command basis
```
$ lastcomm ls
```

Per user basis
```
$ lastcomm username
```

Before we start installing package, please configure Local package mirror so that we can fetch installation files from the local ftp server
```
#vi /etc/csh.cshrc
```
```
setenv PACKAGESITE 'ftp://192.168.0.1/pub/FreeBSD/ports/i386/packages-6.2-release/Latest/'
```

Nmap

Install nmap
```
# pkg_add -r nmap
```
Nmap ping sweep
```
# nmap -sP 192.168.0.0/24 | less
```
Scan for ports
```
# nmap -v 192.168.0.1
```
Os detection and stealth scan
```
# nmap -sS -O -v 192.168.0.1
```
XMas scan
```
# nmap -sX -v 192.168.0.1
```
Output the result to a file
```
# nmap 192.168.0.1 -oN nmap.scan
```

Try nmap using different switches and find out who is the most vulnerable

Nessus

Install Nessus and Nessus Plugins
```
# pkg_add -r nessus-gtk2
# pkg_add -r nessus-plugins
```

It will automatically generate certificate, just enter the appropriate values
```
# /usr/local/sbin/nessus-adduser
```
Enter your desired username
Select Password authentication for Lab exercise
Enter the desired password

Start the daemon
```
# mkdir /usr/local/var/nessus/nessus-services
# /usr/local/sbin/nessusd -D
```

If you want to receive more Plugins you need to register

For regularly update the plugins
Add the following line to /etc/crontab
```
# vi /etc/crontab
```
```
00 3 * * * root   /usr/local/sbin/nessus-update-plugins
```

Run the client
```
% /usr/local/bin/nessus
```

Enter the username and password you created above
Under "target selection" enter the ip address you want to scan for eg 192.168.0.1. Then click on "start the scan" which should take a while depending upon the number of plug ins you have enabled.
Report
You can either view the report in the report windows or save the report as various formats such as ascii, html, xml
Export it to html and it should save a directory named nessus.html. Use a browser to view the report.

TcpDump

Todo this tcpdump pratical please ask the computer next to you to scan your computer and watch the packets.

Trace packets with source Ip from 192.168.0.0/24 network
```
#tcpdump -i rl0 src net 192.168.0.0/24
```

Same as above but without host name translation
```
#tcpdump -i rl0 -n src net 192.168.0.0/24
```

Same as above but without port name translation
#tcpdump -i rl0 -nn src net 192.168.0.0/24

Trace packets with source Ip from 192.168.0.0/24 network and destination 192.168.0.1. "-i any" means listen on all available interfaces, only supported in Linux.
```
#tcpdump -i rl0 -n src net 192.168.0.0/24 and dst 192.168.0.1
```

Trace tcp packets coming in/ going out of interface xl0
```
#tcpdump -i rl0 tcp
```

Trace all packets coming in / going out of interface xl0 that is not arp packets.
```
#tcpdump -i rl0 ! arp
```

Print the packets in both HEX and ASCII
```
#tcpdump -i rl0 -X
```

Save the tcpdump output to a file
```
#tcpdump -w /tmp/tcpdump.out
```

There are lots of other options available with tcpdump

Wireshark

Install Wireshark
```
#pkg_add -r wireshark
```

Run Wireshark
```
%wireshark /tmp/tcpdump.out
```

Wireshark is a nice GUI Network protocol analyzer which is easier to understand than tcpdump. You can read tcpdump output files or if you run wireshark as root you can also capture and display packets in real time. You can also use “Follow TCP stream” to reassemble the data within a particular TCP session. There are lots of other features in Wireshark which can ease network troubleshooting.

Ntop
Install ntop
```
#pkg_add -r ntop
```

Create ntop admin user
```
#/usr/local/bin/ntop -A
```

Start ntop
You can remove the “-d” to see the output on the screen
```
#/usr/local/bin/ntop -u nobody -d
```

Allow ntop from tcpwrapper
```
# vi /etc/hosts.allow
```
```
ntop: 192.168.0. : allow
```

View ntop in your Browser
http://*ntop_server*:3000
Or
```
http://localhost:3000
```

Snort

Install Mysql Server
```
#pkg_add  -r mysql50-server
```

Install snort from ports
```
# pkg_add -r snort
```

Install Apache22 Server
```
# pkg_add -r apache22
```

Install PHP4
```
# pkg_add -r php4
# pkg_add -r php4-mysql
# pkg_add -r php4-session
# pkg_add -r php4-pcre
```

```
#vi /usr/local/etc/apache22/Includes/httpd-php4.conf
```
```
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

Start apache
```
# vi /etc/rc.conf
```
```
apache22_enable=YES
```

```
# /usr/local/etc/rc.d/apache22 start
```

Start Mysql server
```
#vi /etc/rc.conf
```
```
mysql_enable=YES
```

```
# /usr/local/etc/rc.d/mysql-server start
```

Secure Mysql Installation

```
# mysql -u root
drop database test;
use mysql;
delete from columns_priv;
delete from db;
delete from func;
delete from host;
delete from tables_priv;
delete from user where User='';
update user set Password=password('1mysql90admin') where User='root';
flush privileges;
exit
```

Create Database for Snort

```
# mysql -u root -p
create database snort ;
grant all on snort.* to snort@localhost identified by '98kd97';
```

Import snort schema into mysql

```
% mysql -u snort -p snort < /usr/local/share/examples/snort/create_mysql
```

Copy snort sample files

```
# cd /usr/local/etc/snort/
# foreach sample (`ls *-sample`)
echo $sample
cp $sample `echo $sample | sed 's/-sample//'`
end
```

Download Rules File and Extract it to /usr/local/etc/snort/rules/

```
# cd /tmp
#fetch ftp://192.168.0.1/pub/FreeBSD/ports/i386/packages-6.2-
release/Latest/Community-Rules-CURRENT.tar.gz
#tar xzvf Community-Rules-CURRENT.tar.gz -C /usr/local/etc/snort/
# cd /usr/local/etc/snort/
# foreach file (`ls *.rules`)
echo $file
ln -s $file `echo $file | sed 's/community-//'`
end
```

Configure snort to log to mysql database

#vi /usr/local/etc/snort.conf

```
output database:log,mysql,user=snort password=98kd97 dbname=snort
host=localhost
# These files are not present in the Free Rules file so comment them out
#include $RULE_PATH/local.rules
#include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/scan.rules
#include $RULE_PATH/finger.rules
#include $RULE_PATH/telnet.rules
#include $RULE_PATH/rpc.rules
#include $RULE_PATH/rservices.rules
#include $RULE_PATH/ddos.rules
#include $RULE_PATH/dns.rules
#include $RULE_PATH/tftp.rules
#include $RULE_PATH/web-coldfusion.rules
#include $RULE_PATH/web-frontpage.rules
#include $RULE_PATH/sql.rules
#include $RULE_PATH/x11.rules
#include $RULE_PATH/netbios.rules
#include $RULE_PATH/attack-responses.rules
#include $RULE_PATH/mysql.rules
#include $RULE_PATH/snmp.rules
#include $RULE_PATH/pop2.rules
#include $RULE_PATH/pop3.rules
#include $RULE_PATH/other-ids.rules
#include $RULE_PATH/experimental.rules
```

Starting snort

vi /etc/rc.conf

```
snort_enable=YES
snort_flags=”-D -u nobody”
```

```
# chown nobody: /var/log/snort
# /usr/local/etc/rc.d/snort start
# tail -f /var/log/snort/alert
```

Now try to nmap to your machine from other machine and you should see snort detect these attacks.

Analysis Console for Intrusion Databases (ACID)

Install ACID
pkg_add -r acid

```
# vi /usr/local/www/acid/acid_conf.php
```

Update the following fields

```
$DBlib_path = "/usr/local/share/adodb";

$alert_dbname   = "snort";
$alert_host     = "localhost";
$alert_port     = "";
$alert_user     = "snort";
$alert_password = "98kd97";
```

```
#vi /usr/local/etc/apache22/Includes/httpd-acid.conf
```

```
Alias /acid/ /usr/local/www/acid/
<Directory /usr/local/www/acid/ >
        Options None
        Order allow,deny
        allow from all
</Directory>
```

Browse the report Page from a Browser
`http://localhost/acid/`

Go to Setup page
Create ACID AG
Then go to the Main page

ipfw

Enable IPFW in rc.conf

```
# vi /etc/rc.conf
```

```
firewall_enable="YES"
firewall_script="/etc/rc.firewall"
firewall_logging=”YES”
```

Or for Testing

```
# kldload ipfw
```

Insert a test rule

```
# ipfw add allow ip from any to me
```

Listing Firewall Rules

```
# ipfw -a l
```

Delete the rule

```
# ipfw delete 100
```

Allow Rule for ssh

```
# ipfw add allow tcp from any to me 22
```

Deny Rule for telnet

```
# ipfw add deny tcp from any to me 23
```

Deny and Log rules

```
# ipfw add deny log logamount 200 ip from any to any
```

Flush rules

```
# ipfw flush
```

State rules

```
# ipfw add check-state
# ipfw add allow icmp from me to any keep-state
```

Listing Dynamic Rules

```
# ipfw -d l
```

Listing Dynamic expired Rules

```
# ipfw -d -e l
```

Accept all packets from interface loopback

```
# ipfw add allow ip from any to any via lo0
```

Allow incoming connections to port 22 and keep state

```
# ipfw add allow tcp from 192.168.0.0/24 to me 22 in keep-state
```

Allow new connections from this box

```
# ipfw add allow tcp from me to any setup out keep-state
```

Ratelimiting incoming requests to port 80 on the basis of source address
Only allow 10 connections from one host

```
# ipfw add allow tcp from any to me 80 limit src-addr 10
```

Build a firewall

```
# mv /etc/rc.firewall /etc/rc.firewall.old
# vi /etc/rc.firewall
```

```
#!/bin/sh

FW="/sbin/ipfw -q"

#Flush the rules
${FW} -f flush

#Allow from Interface loopback
${FW} add allow all from any to any via lo0

# Check-state
${FW} add check-state

# Allow setup of outgoing TCP connections and keep state
${FW} add allow tcp from me to any setup out keep-state

# Allow udp from this host
${FW} add allow udp from me to any out keep-state

# Allow ssh from local network
${FW} add allow tcp from 192.168.0.0/24 to me 22 keep-state

# Allow icmp
${FW} add allow icmp from me to any keep-state

# Deny echo-request and allow the rest
${FW} add deny icmp from any to me icmptypes 8
${FW} add allow icmp from any to me
```

Make it executable

```
# chmod 755 /etc/rc.firewall
```

Run the firewall

```
# /etc/rc.firewall
```