Global Information Assurance Certification Paper

Build Your Own Security Lab on a Shoestring Budget

Paul Greene

GSEC Version 1.4b – Option 1

Build Your Own Security Lab on a Shoestring Budget

Paul Greene

GSEC Version 1.4b – Option 1

April 7, 2003

1. Abstract

The purpose of this practical is to introduce some basic security concepts to a relatively inexperienced security practitioner and to describe, in "how-to" format, the configuration of a relatively inexpensive "home security lab" setup that can allow a person to practice and experiment with these security concepts in a hands on setting for a relatively small investment of money.

The specific concepts that I'd like to introduce in this paper are:

- firewall technology
- intrusion detection technology
- honeypot technology

All of the software and operating systems in this paper will be open source, and most of the hardware configuration described in this paper was purchased used at local computer shows for a total cost of about $650.

2. Discussion of the basic concepts involved in this paper

Firewall technology

Firewalls control the flow of data between two or more networks. A firewall protects a trusted network from being attacked by an untrusted network; i.e. a firewall can protect the Acme Corporation and all of its valuable corporate data from unauthorized users trying to gain access to its network from the wild and wooly internet.

There are basically three different types of firewalls:

- *simple packet filtering firewalls* – firewalls that restrict the flow of data traffic based on the source and destination addresses contained within the packet, or by the destination port numbers contained within the packet
- *stateful packet inspection firewalls* – firewalls that maintain a table of "state" information about a tcp/ip connection; if a previous connection was allowed through based on having passed an access control list, all subsequent packets of the same connection would pass through; this allows connections initiated by a computer on the "safe" interface of a connection to receive replies back from the "untrusted" side of a connection that would ordinarily be blocked from entering the network
- *proxy firewalls* – proxy firewalls do not allow a direct connection from an untrusted network to connect with another system on the trusted network; a

packet is received on one interface and reviewed according to the security policy, and then passed on to the correct interface or rejected if it does not meet the security policy; "application-level" proxies require a proxy for every service and protocol, and operate at the application layer of the OSI model; "circuit-level" proxies review only packet header information at the network layer of the OSI model, and don't have the detailed control of an "application-level" proxy

Intrusion detection technology

Intrusion detection technology is a passive monitoring technology that watches data traffic for suspicious behavior as it passes through a network ("*network based"*) or an individual host system (*"host based"*). There are two basic types of intrusion detection:

- *signature based* – where information about past intrusions is analyzed and specific bit patterns in the attack are used as "signatures" in recognizing when the same attack is used again
- *statistical or behavior based* – a reference model of behavior that is considered "normal" for the system is compiled, and then anytime behavior falls outside those boundaries of the norm, it is considered "suspicious" network traffic

Intrusion detection systems have generally been "passive" in their approach to network attacks, but newer generation IDS systems are becoming more intelligent. Referred to as "Intrusion Prevention Systems" (IPS), based on certain attacks detected, they can take reactive steps to block further connections from certain network addresses or can make "on the fly" updates to firewall rulesets to prevent further attacks from the same source.

Honeynet/Honeypot technology

Taken from the www.honeynet.org website a "honeypot" is "a system who's value is being probed, attacked, or compromised, usually for the purpose of detection or alerting of blackhat activity"[1]. In other words, a honeypot is a system put on a network with the specific intention of having it attacked by malicious users.

There are two main intentions of installing a honeypot. One is to research the methods that blackhat hackers use in gaining access to systems, and second is to serve as a decoy system that, by being probed and attacked, serves as a warning to network administrators that intruders are active on their network.

The difference between a "honeypot" and a "honeynet" is that a honeypot is a single system, and honeynet is a network.

3. Description of the four systems that make up the security lab

This section will describe the basic configuration of the home security lab.

[1] "Honeynet FAQ", URL: http://www.honeynet.org/misc/faq.html

Definition of requirements

We need to define the requirements of how we want to establish this home security lab.

A few assumptions are made up front:

- The internet connection is a broadband connection (cable or DSL) with a single dynamic IP address assigned
- The connection used as the home security lab is also the same connection that will be used for internet access for the rest of the household
- The "daily use" internet access part of the network needs to be protected from the internet as well as be protected from the subnet that is offering a public service to the internet
- The service being offered to the general internet in this paper is a webserver, although most any other service could be offered in the same way (the main restrictions coming from the ISP providing the broadband access)

The basic overall description of how we want to lay this out is as follows.

Firewall - There will be one computer that makes the actual connection to the ISP and serves as the firewall. This system will run the OpenBSD operating system. This computer will grab a dynamic IP address from the ISP, and through network address translation, will provide the gateway for the rest of the network to gain access to the internet. The firewall will have three separate interfaces. One interface will face the ISP/internet, the second will face the internal protected network, and the third interface will face a DMZ network that will be offering the public facing web service to the internet.

IDS Sensor - A second computer, also running OpenBSD, will have three interfaces, and will serve as the network sensor for the intrusion detection system. Two of the three interfaces will be "stealthy" interfaces that will be on the DMZ network, and one interface will be connected to the internal protected network and the administrative console that will be described in just a minute. The "stealth" interface means that the interfaces are not assigned IP addresses on the DMZ network, but operate at a layer two level on the DMZ network. This system will be invisible to other systems on the DMZ network but will still be able to monitor every packet of traffic that passes through this network. The IDS software on this sensor will be "Snort", the most popular of the open source IDS software.

Webserver - A third computer, running Redhat Linux, will be the system offering the public web service and will act as a honeypot system. An older version of Redhat Linux and the Apache webserver will run on this computer to make it an easier target to attack.

Admin-console - A fourth computer, also running Redhat Linux, will be used as an administrative console. This system will reside on the internal protected network, and

will receive IDS alerts sent from the sensor into a MySQL database, as well as serve as a syslog server to send syslog event logs to. This system will have a nice GUI interface with which to monitor alerts from the sensors

These four computers will make up the home security lab. Any daily use computers used for general internet access will be additional to this list, and located on the internal protected network.

4. Description of the network layout

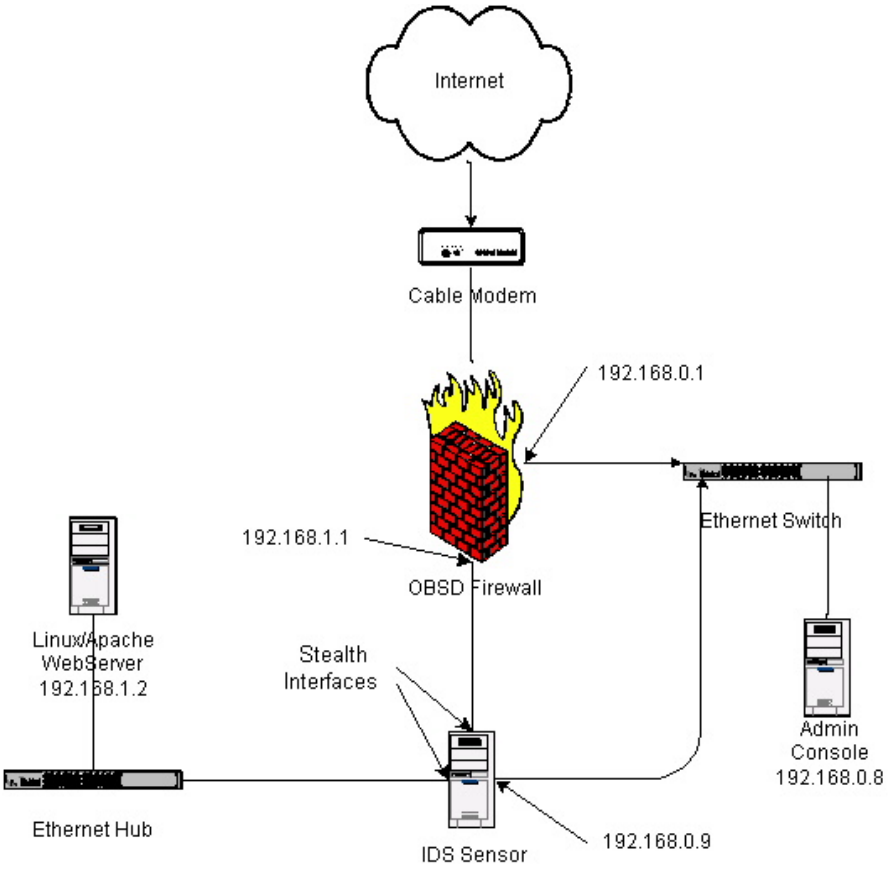
The following Visio diagram gives a graphical overview of the setup being described in this paper.

Note that both the firewall and the IDS sensor have three interfaces each. On the firewall, one interface faces the cable modem and the internet; one interface faces the protected internal network; and one interface faces the DMZ zone.

On the IDS sensor, the stealth interfaces logically reside on the DMZ network, but operate at a layer two level, and thus are invisible to anyone connecting from within the DMZ. The third interface connects to the internal protected network and allows the Snort alerts to be sent from the sensor to the admin-console.

There are different possibilities for sensor placement. The placement of the sensor as described below is monitoring traffic only after it has passed through the firewall. This avoids dealing with the amount of noise generated by “script-kiddie” attacks. A lot of attention given to meaningless alerts can then be avoided, and every alert that is generated in response to an attack should be taken seriously. A limitation of placement here is that it is not monitoring any activity going in and out of the internal protected network. However, because the computers on the internal network are all running personal firewalls and have anti-virus software with the live update feature set to automatically download new virus signatures, plus are protected by rulesets on the firewall blocking traffic before it even enters the internal network, they are already quite well protected.

A sensor could also have been placed between the cable modem and the firewall. The advantage of placement here would be that all traffic entering and leaving both the internal protected network and the DMZ network would be monitored by the IDS. The drawback of placement in this location would be the large amount of mostly useless alerts generated by “script-kiddie” attacks. Also, another drawback would be that because the IDS would be sniffing packets before having gone through NAT, the destination IP address would still be the external interface of the firewall, so it would be more difficult to figure out the true destination of the packet.



5. Description of the Open Source components used in this network

OpenBSD operating system

`http://www.openbsd.org`

OpenBSD was chosen as the operating system for the firewall and the IDS sensor because of its sterling record for security. In the book *"Security Architecture – Design, Deployment, and Operations"* by RSA Press[2], they make the statement "OpenBSD is the closest to a secure out-of-the-box OS distribution available today". The OpenBSD project team has done an exhaustive operating system source code audit for the past seven years, and has made a proactive effort to seek out potential bugs and security defects in the operating system. The result is a remarkably bug free, secure, and stable operating system that has had just a single remotely exploitable hole in the default operating system install in 7 years.

Redhat Linux operating system

[2] King, Christopher M., Dalton; Curtis E.; & Osmanglu, T. Ertem. "Security Architecture – Design, Deployment, & Operations". 2001. RSA Press. p. 259

```
http://www.redhat.com
```

Linux has been the most popular operating system of the open source software movement for many years now. Because of it's popularity it tends to be better supported as far as available applications, so Linux will be used for the administrative console and for the web server.

Snort Intrusion Detection System

```
http://www.snort.org
```

Snort will be installed on the IDS sensor and the administrative console. Snort is the most popular of the open source intrusion detection systems, and has a very active development team continually improving the product, and updating the file attack signatures. Also Snort has many plugins that can be added to increase its functionality, including GUI front ends to make monitoring the alerts and system logs easier. Snort can also run in packet capture mode and capture the full content of every packet passing through the system.

Tripwire

```
http://sourceforge.net/projects/tripwire
```

The webserver is going to be unprotected from attack from the internet. Since part of the intention of this webserver is to act as a honeypot, it is hoped that it will eventually be broken into. Tripwire keeps watch on the important files on the system and will warn you if any of these files are changed in any way.

chkrootkit

```
http://www.chkrootkit.org
```

This software will be installed on the webserver. "chkrootkit" will examine the system for the installation of any installed rootkits that are likely to be installed on the web server if it is ever compromised.

tcpdump

```
http://www.tcpdump.org
```

"tcpdump" is a packet capture software that can be used for sniffing packets off a network, and can also be used to read Snort packet captures that have been saved in a binary format.

Ethereal

```
http://www.ethereal.com
```

Ethereal is another excellent packet capturing software available for Linux or Windows that also has the advantage of a user-friendly GUI front end.

6. Building the firewall

Operating system installation

Installing with a CD-ROM is probably the easiest way to do an install of OpenBSD. Buying the CD also helps to support the OpenBSD project, since the project is not a commercial venture.

Personally though, I generally use the ftp install of OpenBSD. One reason is that I'm located only a couple of hops away from a major internet backbone (UU.net) that also serves as a mirror site for OpenBSD. Ftp installs always assure you that you're getting the latest available version.

The OpenBSD project has a very good installation FAQ at the following URL:

```
http://www.openbsd.org/faq/faq4.html
```

I don't want to repeat the information that is contained in good sources already. I will instead attempt to fill in the "gotcha" parts where you'd have to go to multiple sources of information to figure out how to do something.

A few gotchas with respect to the installation:

Since this is a firewall, and does not have a need for many users, you can significantly reduce the /home partition from the large size recommended in the install FAQ; you shouldn't have more than 2 or 3 regular user accounts, so you could even eliminate /home as a separate partition altogether if you want.

You need a lot of space for logging, so you can significantly increase the size of the /var partition to accommodate a large amount of logs.

When you reach the part of the install where it asks you if you want to configure the network, if you are connecting the firewall box directly to your cable modem, choose "dhcp" when it asks you what IP address to assign to the interface; the ISP DHCP server should then provide you with an IP address, a default gateway, and a DNS server IP address; if you are installing from within an existing network, then you should already be familiar with your gateway address, DNS servers, and what IP address you want to assign to this particular box.

When you reach the part where it asks you:

```
Sets can be located on a (m)ounted filesystem; a (c)drom, (d)isk or (t)ape
device; or a (f)tp, (n)fs or (h)ttp server.
```

```
Where are the install sets you want to use? (m, c, f, etc.)
```

Pick “f” for the FTP install. You will see a display that you can scroll through to find the international mirror sites that are closest to you.

When you get to the part in the install where it asks you what disk sets you want to install, skip the games, and all of the x-windows related packages; a firewall should be stripped down to only the bare essentials.

Configuration of the firewall

After the installation is done running, the system will reboot. Now the fine tuning will begin!

Type the following command to view the interfaces recognized by OpenBSD.

```
ifconfig –a
```

This should show all the interfaces recognized by the operating system. In the configuration described in this paper there are three interfaces. Each interface will have an entry similar to the following. (Note the “dc0” in the beginning.)

```
dc0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        address: 00:00:f8:d9:a9:0f
        media: Ethernet none (100baseTX)
        status: active
        inet6 fe80::200:f8ff:fed9:a90f%dc0 prefixlen 64 scopeid 0x1
        inet <ip.address> netmask 0xffffff00 broadcast <broadcast.ip>
```

On this particular system, the three interfaces listed are “dc0”, “rl0”, and “ep0”. The first two letters of the interface depends on the driver used, and the number is the number of the interface, starting with “0”. If there’s more than one interface of the same driver, the second interface will be numbered “1” and so on. Unless your system is using the exact same network cards as in this description (not likely!), the interface names listed will be different than the above. Substitute the names according to what’s actually in your system.

The layout of the interfaces will be as follows:

- dc0 is the interface that connects to the cable modem, and will get its IP address assigned dynamically by the ISP
- ep0 is the interface that connects to the internal protected network, and will have a static ip address of 192.168.0.1 with a netmask of 255.255.255.0
- rl0 is the interface that connects to the DMZ network, and will have a static IP address of 192.168.1.1 with a netmask of 255.255.255.0

Each interface needs a configuration file in the `/etc` directory. If you configured any interfaces during the install, the configuration file for those interfaces should have already been created.

The naming convention of the configuration file is in the following format. "`hostname.<interface_name>`".

1. Type the following commands to manually create each configuration file.

```
cd /etc
touch hostname.dc0
touch hostname.ep0
touch hostname.rl0
```

2. The following lines insert the configuration information in each file. (Note: the word NONE needs to be capitalized)

```
echo 'dhcp NONE NONE NONE' > hostname.dc0
echo 'inet 192.168.0.1 255.255.255.0 NONE' > hostname.ep0
echo 'inet 192.168.1.1 255.255.255.0 NONE' > hostname.rl0
```

3. Make a backup copy of the "dhclient.conf" file with the following command:

```
cp dhclient.conf dhclient.conf.bak
```

4. Open the "dhclient.conf" file with vi, and paste in the following information after the commented lines in the file:

```
initial-interval 1;
interface "dc0" {
send host-name "<hostname>";
request subnet-mask, broadcast-address, routers,
        domain-name, domain-name-servers, host-name;
```

   This file is based on the configuration required for Comcast cable service but should work for other cable modem services as well. Substitute the correct interface name for "dc0", and <hostname> is the hostname assigned by the ISP. The quotation marks in the file are **supposed** to be there, so don't leave them out.

5. Open the "`rc.conf`" file using vi. Find the line that says "`pf=NO`" and change it to "`pf=YES`".

6. Open the "`sysctl.conf`" file using vi. Uncomment the line that says "`net.inet.ip.forwarding=1`".

7. The following command will delete the original firewall rules configuration file, "`pf.conf`", but keep a backup copy named "`pf.conf.bak`":

```
mv pf.conf pf.conf.bak
```

8. Create a fresh new empty "`pf.conf`" file with the following command:

```
touch pf.conf
```

   A complete and heavily commented "`pf.conf`" file is listed in Appendix A. It is designed to be an easy drop in for anyone building the setup described in this paper. Only the references to the three interfaces need to be edited to suit the specific interfaces used by the reader.

Another useful reference with a few other ideas about configuration of OpenBSD as a layer two device can be found at:

```
http://ezine.daemonnews.org/200207/transpfobsd.html
```

As a general practice, an administrator should first logon using a non-privileged user account and then use the "`su`" command to switch to root. In order to use the "`su`" command, the user account must first be added to the wheel group. Edit the file "`/etc/group`", and add the usernames that need root level access to the "`wheel`" group.

This finishes the configuration of the firewall.

7. Building the IDS sensor

Operating system installation

The operating system installation of the IDS sensor is the same as for the firewall, so I don't want to repeat the same information here. I will just jump right into the interface configuration.

As mentioned earlier, the sensor will have two interfaces configured in a bridge configuration that will invisibly monitor the DMZ subnet, and one interface will be assigned as an administrative interface connected to the internal network.

Running the command "`ifconfig -a`" on the sensor reveals three interfaces; "dc0", "rl0", and "xl0". The way we are going to lay out these interfaces is as follows:

- "dc0" is the administrative interface that will be connected to the internal network; in this particular configuration, the statically assigned IP address will be 192.168.0.9, with a netmask of 255.255.255.0
- "rl0" will be assigned to the bridge interface and will not be assigned an IP address
- "xl0" will also be assigned to the bridge interface and will not be assigned an IP address

1. The configuration files for each interface are created.

```
cd /etc
touch hostname.dc0
touch hostname.rl0
touch hostname.xl0
touch bridgename.bridge0
```

2. The following commands will configure the appropriate settings for each interface.

```
echo 'inet 192.168.0.9 255.255.255.0 NONE' > hostname.dc0
echo 'up' > hostname.rl0
echo 'up' > hostname.xl0
echo 'add rl0 add xl0 up' > bridgename.bridge0
```

3. Open the "`rc.conf`" file using vi. Find the line that says "`pf=NO`" and change it to "`pf=YES`".

4. Open the "`sysctl.conf`" file using vi. Uncomment the line that says "`net.inet.ip.forwarding=1`".

5. After making these changes, reboot the system, and it should come up with all interfaces automatically activated.

The sensor will not be implementing any firewall rules in the "`pf.conf`" file, so this file will not be activated. By default, every sample line in "`pf.conf`" is commented out.

Again, with the sensor, it is recommended to not log directly on to the system as root, but to logon with a regular user account, and then use the "su" command to switch to root. Add the appropriate usernames to the "`wheel`" group in "`/etc/group`".

Ethernet cabling

In our network layout, the stealth interfaces on the sensor are being used like a bridge between the firewall and the 4 port hub. The ethernet cable that runs between the firewall and the IDS sensor must be a **crossover** cable, not a regular CAT 5 ethernet cable. The ethernet cable that runs from the other interface to the hub is a regular CAT 5 ethernet cable.

Snort Installation

1. Earlier versions of Snort used to have precompiled binaries for OpenBSD. Current versions now have precompiled binaries only for Linux and Windows. Download the latest source code from the Snort website at:

```
http://www.snort.org/dl/
```

2. Copy the downloaded source code to the "`/usr/src`" directory. Then run the following command to extract the source code files.

```
tar –zxvf snort-1.9.1.tar.gz
```

3. This extracts the source code files to a directory named "`snort-1.9.1`". Run the following command to switch to the snort directory.

```
cd snort-1.9.1
```

4. To compile snort with support for logging to a MySql database, run the following commands:

```
./configure --with-mysql
make
make install
```

5. Download the latest rulesets for snort from the following address:

```
http://www.snort.org/dl/rules/
```

   Be sure to download the rules marked "`stable`" and not "`current`".

6. Create a directory in which to place the snort rules. Run the following commands to create the directory, copy the downloaded file to that directory, and untar the snort rules.

```
mkdir /etc/snort
cp snortrules-stable.tar.gz /etc/snort
cd /etc/snort
tar –zxvf snortrules-stable.tar.gz
```

7. The following commands move the rules into the "`/etc/snort`" directory

```
cd /etc/snort/rules
mv * ../
cd ..
rmdir rules
```

8. The "`snort.conf`" file holds the configuration settings for snort. We need to tweak some of the settings to point to the MySQL database server on the admin-console.

```
vi snort.conf
```

9. Scroll down into the "`snort.conf`" file and uncomment the following line:

```
output database: log, mysql, user=root password=test dbname=db
host=localhost
```

10. To coincide with the configuration of the admin-console that we will be building next, set the parameters in the above line to the following:

```
user=snort
password=snort
dbname=snort
host=<ip.address.of.admin-console>
```

11. Comment out the line in "`snort.conf`" that says "`var RULE_PATH ../rules`", and then remove the character string "`$RULE_PATH/`" from each line of the rulesets.

    For example, change the line:

```
include $RULE_PATH/bad-traffic.rules
```

    To the following:

```
include bad-traffic.rules
```

12. The snort logging directory should be created with the following command:

```
mkdir /var/log/snort
```

Changes made to syslog

The admin-console is also going to serve as a syslog server. The following tweaks will enable syslog to also send its logfiles to the syslog server/admin-console. Uncomment the following lines, and change the "`@loghost`" variable to "`@<ip.address.of.admin-console>`".

```
*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none     @loghost
kern.debug,user.info,syslog.info                            @loghost
auth.info,authpriv.debug,daemon.info                        @loghost
```

Note: Blank white space in the "`syslog.conf`" file must be tab characters. Don't use the spacebar for any separation between the parameters.

8. Building the Administrative Console

The administrative console will be running the Redhat Linux operating system. The Snort project and the Honeynet project both seem to favor Linux as their operating system of choice, so support for the function of the administrative console will be easier using Linux.

The installation of the Redhat Linux, MySQL, the ACID console, and the web-based interface is quite lengthy. It's not especially complicated, but to describe everything in detail would not fit within the limitations of this paper.

For this reason I would like to refer the reader to two documents that do an excellent job of describing installing Linux, MySQL, the ACID console, and the web interface. In this paper I will deal with the "gotchas" that might hang up the installation, and also describe some other tweaks that are needed to improve the functionality.

The first one is authored by Steven J. Scott and is available off of the Snort website:

```
http://www.snort.org/docs/snort-rh7-mysql-ACID-1-5.pdf
```

The second is authored by Anton Chuvakin and Vladislav V. Myasnyankin and is available off of the Security Focus website:

```
http://www.securityfocus.com/infocus/1640
```

I would recommend using the Steven J. Scott guide for the step-by-step installation, and use the other guide as bonus material.

Operating system install

The Steven J. Scott document installs Redhat 7.3 and Snort 1.8.7 in its example. For the sake of this document, the current version of Redhat is 8.0, with 9.0 due to be released shortly; the current version of Snort available is 1.9.1, with version 2.0 due to be released any day now. For the purposes of this paper Redhat 8.0 and Snort 1.9.1 seemed to install just fine using the Steven J. Scott instructions. Version 2.0 of Snort is supposed to have some more extensive changes in this version upgrade, so it will probably take more tweaking of the install instructions to get things into a functioning state.

As described in the OpenBSD installation of the firewall and the IDS sensor, the admin-console also does not need much space for user home directories ("`/home`"). The "`/var`" partition does need to be quite large however, and the Steven J. Scott guide recommends reserving several gigabytes worth of space for this partition.

In the network configuration, the install guide says to configure "`eth1`" as a sniffing interface. However, in the configuration as described in this document, the admin-console is **not** being used as a sniffer; that is the purpose of the IDS sensor. The admin-console only has one NIC interface installed on it.

Additional items to install on the admin-console

Nmap

There are a couple of additional tools that can be added to the admin-console to make it more useful. One of them is called "Nmap", which is a network utility for network exploration, security auditing, port scanning, and remote OS detection.

This is useful for keeping watch on the webserver and firewall described earlier in this paper. If all of a sudden new ports are opened up on the webserver or the firewall, this is a very good indication that something fishy is going on.

Nmap also has a GUI front end that can be installed to make it more user friendly.

Nmap is included on the Redhat installation CDs but the version number is usually out of date by one or two releases.To install the most recent version of both the Nmap program and the GUI front end, once your network is up and running and you have internet access from the admin-console, you can run the following commands, and Nmap will install automatically using a web install.

```
rpm -vhU http://download.insecure.org/nmap/dist/nmap-3.20-1.i386.rpm
rpm -vhU http://download.insecure.org/nmap/dist/nmap-frontend-3.20-1.i386.rpm
```

If you need to download Nmap first and then install, download the RPMs from the following URL:

```
http://www.insecure.org/nmap/nmap_download.html
```

And then run the following commands:

```
rpm -ivh nmap-3.20-1.i386.rpm
rpm -ivh nmap-frontend-3.20-1.i386.rpm
```

Ethereal

Ethereal is a network protocol analyzer that allows you to sniff traffic off a live network, or read packet capture files created by tcpdump or Snort (if created with the binary format switch). Ethereal is extremely useful when performing packet analysis because it is very easy to read the contents of each packet, and you can sort the data in useful ways (i.e. follow a specific data stream, sort by protocol, etc).

Ethereal is included with the Redhat installation CDs, so it would be easier to install it off of the CDs using the command:

```
rpm –ivh ethereal*.rpm
```

The latest and greatest version of Ethereal can always be found at RPMFind.net:

```
http://rpmfind.net/linux/RPM/EByName.html
```

9. Building the webserver

An older version of Redhat and Apache was intentionally installed on the webserver in order to make the server easier to attack. No effort was made at hardening the system or implementing any security controls beyond that given by the default installation. This

is useful for a person that would like to experiment with the concept of Honeypots. However, if security for the webserver is of concern, it would be highly recommend to either apply a hardening process to both the Linux operating system and the Apache webserver, or configure the webserver using OpenBSD, which already installs the operating system and the Apache webserver secure by default.

The installation of the operating system on the webserver is the easiest of any system in our configuration. Select the “Server” installation during the Redhat installation program, choose “Webserver” during the options section, and go with all of the default settings. Installing Xfree86 is not needed for this installation, though if the user prefers a GUI interface, go ahead and install it anyway.

In the configuration as described in this paper, when you reach the section on network configuration the following settings would apply:

```
static ip address of 192.168.1.2
gateway of 192.168.1.1
dns servers – <depends on your ISP>
choose the option – “activate at boot”
```

Adding content to the webserver

For the purpose of this paper, the actual content of the webserver does not matter. A simple personal web page is fine, or if you want to really go to town and create something more elaborate, then knock yourself out. In a default Apache configuration on Linux, the web content goes in the following directory:

```
/var/www/html
```

The main file for the web page should be named “`index.html`”.

The important extras for the webserver

Because the webserver will be an object of attack for malicious attackers, two important pieces of detective software are important to install in order to help discover signs of possible system compromise. One is called “`tripwire`”, which can detect changes in important system and configuration files, and the other is called “`chkrootkit`”, which can search for the presence of known rootkits that are likely to be installed in the event of a successful system compromise.

Tripwire

Tripwire is included on the Redhat Linux installation CDs though it is not installed by default. The latest and greatest version can also be downloaded at the following URL:

```
http://www.rpmfind.net/linux/RPM/TByName.html
```

1. To install the RPM file, run the following command:

   ```
   rpm –ivh tripwire-2.3-55.i386.rpm
   ```

   Important note: Tripwire must be installed and initialized before being connected to the network. Otherwise you run the risk of creating a tripwire baseline after the system has already been compromised.

2. A useful file to review before proceeding with the installation is found at:

   ```
   /usr/share/doc/tripwire-2.3/quickstart.txt
   ```

3. After running the RPM installation, tripwire must still be configured to actually run. The RPM copies an install script named "`twinstall.sh`" to the "`/etc/tripwire`" directory by default. Run the following command to activate the install script:

   ```
   /etc/tripwire/twinstall.sh
   ```

   You will be prompted to create a password at 2 or 3 different points as the script is processing, during the creation of some cryptographic keys.

4. The tripwire database must be initialized as the first step. Run the following commands to initialize the database:

   ```
   /usr/sbin/tripwire --init
   /usr/sbin/tripwire --check
   ```

5. This will generate a long list of files that tripwire complains is not found on the system, because, surprise!, they are not installed on the system. Print out a copy of the generated report and then compare it to the list of files that tripwire looks for in "`/etc/tripwire/twpol.txt`". Comment out all the files in your generated list from "`twpol.txt`".

6. The file "`/etc/tripwire/twcfg.txt`" is the other configuration file for tripwire. This file sets up the default directories and file locations, and would normally not need to be changed.

7. After making the necessary changes to "`twpol.txt`" (and "`twcfg.txt`" if needed), run the database initialization over again.

8. By default, tripwire installs itself as a daily cron job and mails the results to the root user. A copy of the report is saved in:

   ```
   /var/lib/tripwire/report/
   ```

9. In the event the system is compromised, all of the tripwire keys and configuration files should be considered suspect. Keep a backup copy of the following

directories on a secure server; burn them to CD-ROM, or save them on a write-protected floppy disk. Periodically compare these backups to the ones on your system.

```
/etc/tripwire
/var/lib/tripwire
```

chkrootkit

In the event that a blackhat attacker compromised the webserver, in all likelihood they are going to install a rootkit on your system. This program, "`chkrootkit`", examines your system for the presence of 44 different rootkits and worms, and also checks important system files for signs of tampering.

1. Download the sourcecode for chkrootkit from the following website:

   ```
   www.chkrootkit.org
   ```

2. Copy the "`chkrootkit.tar.gz`" to the "`/usr/src`" directory, and then run the following command to untar and unzip it:

   ```
   cd /usr/src
   tar –zxvf chkrootkit.tar.gz
   ```

   This will create a subdirectory under "`/usr/src`" named "`/usr/src/chkrootkit-0.39a`".

4. Type the following commands to compile the program:

   ```
   cd chkrootkit-0.39a
   make sense
   ```

5. Type the following command to run the program:

   ```
   /usr/src/chkrootkit-0.39a/chkrootkit
   ```

6. This will generate a short list of false positives under "suspicious files and directories". Since chkrootkit should have been installed on the server before connecting it to the internet, it hasn't had a chance yet to be infected by a rootkit. The first time the program is run, make a note of the directories that it flags, and keep that information for future reference.

10. Start doing something with it

With the equipment described thus far in this paper, there are a lot of things that can be done to get hands on experience in the areas of firewalls, intrusion detection, and honeypots.

Firewalls

OpenBSD has a lot more options to work with as a firewall than just what is described in this paper. OpenBSD can be configured for VPN tunneling, and also has integrated hardware support for numerous cryptographic devices, support for Kerberos IV and V, a free ISAKMP daemon, integrated IPSec, and numerous other cryptographic functions. Since the OpenBSD project is based in Canada, there are no export restrictions on any of the cryptography included as part of the operating system.

Possible projects to experiment with OpenBSD as a firewall could be:

- Configure VPN access from a remote computer to your home computer.
- Build a better and more secure authentication system using integrated Kerberos.
- Experiment with the implementation of IPSec

Intrusion Analysis

Snort can be run in sniffer mode that will grab the entire contents of a packet off the wire, and be reviewed later at your leisure with snort, tcpdump, or Ethereal.

For example, if you run the following command:

```
snort –b –L testfile –i bridge0
```

It will save all packets passing on the wire in a tcpdump binary format (the "-b" switch), to a file named "testfile" (the "-L" switch), that are passing through the bridge0 interface (the "-i" switch). If a directory is not specified, the file will be saved in the "`/var/log/snort`" directory, and the given name plus an appended time stamp.

Stop the sniffing by entering a "`Control – c`".

The file can then be reviewed with snort using the following command:

```
snort –r testfile.1049335105
```

This packet analysis can be a very useful experiment. You could log the standard tcp 3 way handshake on a normal connection, log a syn flood attack (that manipulates the standard 3 way handshake), and then compare the two data streams. What's different between the two?

Download some blackhat hacking tools off the internet. Run them against your own systems and log all the interaction. Compare them to what is supposed to happen in a normal TCP/IP traffic flow.

(For a better understanding of TCP/IP theory, refer to the classic book "TCP/IP Illustrated" by W. Richard Stevens. Although the book is a bit old, the last publishing date was 1994, it is still considered one of the best books around on TCP/IP theory.)

Honeypots

Everything needed for getting your feet wet with honeypot technology is already installed with the configuration described in this paper. Monitor the logs on the IDS sensor daily; monitor the webserver access and error logs in "`/var/log/httpd`" daily also.

The Honeynet project is constantly updating its honeypot tools. Keep an eye on their tools web page to stay up to date with bridging utilities, startup scripts, data capture utilities, etc.[3]

11. References

All of the products referred to in this paper have great technical resources to turn to when help is needed.

Mailing Lists

Redhat Linux
https://listman.redhat.com/mailman/listinfo/redhat-list

OpenBSD
Send request to "majordomo@openbsd.org" with message body of "subscribe misc"

Snort
http://lists.sourceforge.net/lists/listinfo/snort-users

Honeynets, Pen Testing, IDS, Security Basics, more
http://www.securityfocus.com/archive

Configuration Guides

Scott, Steven J; "Snort Installation Manual – Snort, MySQL and ACID on Redhat 7.3". Version 1.5. August 2002. URL: http://www.snort.org/docs/snort-rh7-mysql-ACID-1-5.pdf

Chuvakin, Anton & Myasnyankin, Vladislav V. "Complete Snort-based IDS Architecture". November 6, 2002. URL: http://www.securityfocus.com/infocus/1640

Websites

[3] Honeynet Project. "Tools for Honeynets" web page. http://www.honeynet.org/papers/honeynet/tools/ April 7, 2003

http://www.redhat.com
http://www.openbsd.org
http://www.apache.org
http://www.snort.org
http://www.honeynet.org
http://www.chkrootkit.org
http://www.nmap.org
http://sourceforge.net/projects/tripwire
http://www.mysql.com
http://www.rpmfind.net/linux/RPM
http://www.ethereal.com
http://www.cisecurity.org

Books

The Honeynet Project. “Know Your Enemy – Revealing the Security Tools, Tactics, and Motives of the Blackhat Community”. October 2001

King, Christopher M., Dalton; Curtis E.; & Osmanglu, T. Ertem. “Security Architecture – Design, Deployment, & Operations”. 2001. RSA Press

Stevens, W. Richard. ”TCP/IP Illustrated, Volume 1 – The Protocols”. Addison Wesley. 1994

APPENDIX A

The following content should be able to be just cut and pasted into the blank pf.conf file, and should work right away with just minor editing of the interface names in the very beginning of the file. This file is heavily commented in order to help the reader understand the purpose and reasoning behind each line. The firewall rules in OpenBSD have a lot more sophisticated options that can be implemented; this one is kept simple to make it more understandable and easy to follow.

```
# The configuration of this firewall includes 3 interfaces
#
# The first interface (dc0) is connected to the cable modem connection and
# gets its IP address from a DHCP server at the ISP.
#
# The second interface (ep0), connected to the internal, protected network,
# uses a static IP address from the 192.168.0.0/24 network.
#
# The third interface (rl0) is connected to a DMZ where the all the systems
# running public services are located. This interface is using
# a static IP address from the 192.168.1.0/24 network.
#
# The next line sets up the interface names as a variable, simplifying
# administration if any changes had to be made in the future. Simply
# edit the following three lines to match the appropriate interface
# names on your system and the rest of the configuration should work.

ext_int = "dc0"
dmz_int = "rl0"
int_int = "ep0"

# The first command used is the "scrub" command.
#
# The "scrub" command reassembles fragmented packets and normalizes
# incoming packets.

scrub in all

# The next two lines configure network address translation (NAT). NAT
# allows a single IP address given by an ISP to be shared among an
# entire
# internal network. When a system on the internal network makes a
# connection to the internet, NAT converts the internal IP address
# to make it appear as though it came from the IP address assigned by
# DHCP from the ISP to the external interface on the firewall. NAT
# creates a
# table of mappings between the internal IP addresses and source port
# numbers, with the IP address of the external interface, so when
# a response is received, NAT knows which internal IP address the
# incoming packet should be directed to.
#
# The following two lines setup NAT between the two internal networks
# and the "outside world".

nat on $ext_int from 192.168.0.0/24 to any -> $ext_int
```

```
nat on $ext_int from 192.168.1.0/24 to any -> $ext_int

# The "redirect" command enables a request for a service to be
# redirected from the firewall to a server on the internal network
# Since it would be considered bad practice to have a webserver or
# mailserver running on a firewall, yet there is only a single IP
# address given by the ISP, redirection enables multiple servers
# to be accessible from a single IP address. The following rule
# redirects requests for a web server to a server in the DMZ
# internal network with an IP address of 192.168.1.2 using the
# standard http port number of 80.

rdr on $ext_int proto tcp from any to $ext_int port 80 -> 192.168.1.2 port 80

# The following command sets up anti-spoofing to prevent spoofed packets from
# entering or leaving the network. In OpenBSD, a single rule can substitute
# for numerous lines of rules in a typical firewall ruleset.

antispoof for $ext_int

# The next rule sets the default deny policy for the firewall. In
# OpenBSD, there is a policy of "last matching rule applies". In other
# words, you set the default policy in the beginning, and then create rules
# to allow exceptions to the default policy afterwards.

block in on $ext_int all

# We want to allow secure shell access to the firewall for
# administrative purposes. With the "keep state" option, when a
# new connection is initialized, the firewall keeps track of the
# "state" of the connection, and then all subsequent packets
# related to that connection are passed without evaluating
# them according to any more rules. A security feature related
# to the "keep state" option is that it implements sequence
# number checking and can help prevent packet spoofing.
#
# In plain english, the following rule states "allow any packets coming in
# on the external interface, from any source address, and using the tcp
# protocol, and a destination port of 22" (the ssh port).

pass in on $ext_int proto tcp from any to $ext_int port 22 keep state

# The following rule allows any packets coming in on the external interface,
# from any source address, destined for a tcp port number 80, to be passed
# through to the IP address of the webserver at 192.168.1.2. Because of the
# redirect rule created earlier, the http traffic is redirected to the server
# on the internal network.

pass in on $ext_int proto tcp from any to 192.168.1.2 port 80 keep state

# The following rule allows icmp traffic so that the firewall will
# be able to respond to pings and traceroute commands. In many cases,
# you would normally want to block these responses, but in our case,
# we are making it a little bit easier for potential attackers to at
# least find us.

pass in on $ext_int inet proto icmp all icmp-type 8 code 0 keep state
```

```
# The following rule allows unrestricted outgoing traffic. It allows
# internal computers to make any outgoing connection
# as long as the connection is initiated from the internal computer.

pass out on $ext_int all keep state

# The following rules set up a free flow of traffic on the DMZ interface.
# Remember, "in" and "out" is relative to the firewall itself, so
# "outgoing" traffic on this interface refers to traffic going from
# the firewall to the DMZ internal network, and "incoming" refers to
# traffic coming into the firewall from the DMZ network.

pass out on $dmz_int all keep state
pass in on $dmz_int all keep state

# With the rules set up above, the DMZ network would have free access
# to the internal protected network, so we need to setup protection
# to safeguard the protected network. Computers on the internal protected
# network will still be able to access the DMZ.
#
# The first rule allows all traffic leaving the protected internal
# network to pass into the firewall.

pass in on $int_int all keep state

# This next rule sets up a default deny rule for all traffic passing
# into the protected network originating from the internet or from the DMZ.

block out on $int_int all

# This next rule creates an exception to the default deny rule so that
# it is possible to access the internal network from the firewall only.
# By using the IP address itself and a 32 bit subnet mask on both the
# "from" and "to" IP addresses, it allows access to the internal network
# from only the IP address of the firewall.

pass out on $int_int from 192.168.0.1/32 to 192.168.0.6/32

# If you want to be able to access the entire internal network from the
# firewall, then comment out the above line and uncomment the line below

# pass out on $int_int from 192.168.0.1/32 to 192.168.0.0/24

# end of pf.conf file
```

Last Updated: May 29th, 2017

Upcoming Training

Community SANS Ottawa SEC401	Ottawa, ON	Jun 05, 2017 - Jun 10, 2017	Community SANS
SANS San Francisco Summer 2017	San Francisco, CA	Jun 05, 2017 - Jun 10, 2017	Live Event
Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS Charlotte 2017	Charlotte, NC	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Rocky Mountain 2017 - SEC401: Security Essentials Bootcamp Style	Denver, CO	Jun 12, 2017 - Jun 17, 2017	vLive
SANS Secure Europe 2017	Amsterdam, Netherlands	Jun 12, 2017 - Jun 20, 2017	Live Event
Community SANS Portland SEC401	Portland, OR	Jun 12, 2017 - Jun 17, 2017	Community SANS
SANS Rocky Mountain 2017	Denver, CO	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Minneapolis 2017	Minneapolis, MN	Jun 19, 2017 - Jun 24, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS Cyber Defence Canberra 2017	Canberra, Australia	Jun 26, 2017 - Jul 08, 2017	Live Event
SANS Paris 2017	Paris, France	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
Cyber Defence Japan 2017	Tokyo, Japan	Jul 05, 2017 - Jul 15, 2017	Live Event
SANS Cyber Defence Singapore 2017	Singapore, Singapore	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Minneapolis SEC401	Minneapolis, MN	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS Los Angeles - Long Beach 2017	Long Beach, CA	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Phoenix SEC401	Phoenix, AZ	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS Munich Summer 2017	Munich, Germany	Jul 10, 2017 - Jul 15, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Jul 12, 2017 - Sep 13, 2017	Mentor
Mentor Session - SEC401	Macon, GA	Jul 12, 2017 - Aug 23, 2017	Mentor
Community SANS Atlanta SEC401	Atlanta, GA	Jul 17, 2017 - Jul 22, 2017	Community SANS
Community SANS Colorado Springs SEC401	Colorado Springs, CO	Jul 17, 2017 - Jul 22, 2017	Community SANS
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC401: Security Essentials Bootcamp Style	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
Community SANS Charleston SEC401	Charleston, SC	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Fort Lauderdale SEC401	Fort Lauderdale, FL	Jul 31, 2017 - Aug 05, 2017	Community SANS
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event