11 step guide to build a Debian based Intrusion Detection Sensor (IDS) with Snort 2.4.5 or Snort 2.6

By Andy Firman
Last update: June 23, 2006

With new versions of Snort, Base, and Oinkmaster just released, I thought it would be a great time to make a step-by-step guide on Debian!

The most current version of this guide will be at: http://firmanix.com/deb-snort-howto.pdf

Snort is the most widely deployed intrusion management technology worldwide. It is capable of packet sniffing, packet logging, and network-based intrusion detection. Snort has four primary components. The first is the packet sniffer and decoder which reads the datagrams off the wire using the libpcap library. The second component is a preprocessor which is a plug-in that examines the data for things such as malformations, anomalies, and non-compliance and then passes the data off to the detection engine for further inspection. This detection engine is the third component of snort and is most likely what comes to mind when one thinks of the Snort program. The detection engine takes the normalized and stream re-assembled data from the preprocessors and inspects this traffic against the rule base. Finally, the output and alerting module handles the big job of writing and logging packets and alerts in various ways according to one's configuration.

Snort 2.6 has recently been released with many new features, some of which are:

* Added Performance Profiling Measurements for rules & preprocessors.
* Added support for dynamically loadable preprocessors, detection engine and rules.
* Addition of dynamically loadable SMTP and FTP/Telnet preprocessors.
* Preprocessor configuration validation.
* Stream API to simplify transition to next generation Stream module.
* Logging of Generator ID to MySQL database.

Debian Testing (Etch) is always in a state of change with new software being introduced frequently. If you find something changes and affects this guide, please notify me at andy@firmanix.com.

Table of Contents:

1. Debian Testing (Etch) install and related software

First perform a Debian Net Install. Get the iso here: http://www.debian.org/devel/debian-installer/
To install Debian testing, we recommend you use the etch beta 2 release of the installer,
after checking its errata. The following images are available for beta 2:
Choose this for i386 * netinst CD image (100-150 MB) [i386]
Burn the iso image: debian-testing-i386-netinst.iso

Start the install and when you get to the sources section,
choose http and then choose ftp.us.debian.org (default for USA)
and it will begin to download the package listings.

When you get to the "Debian Software Selection" screen,
just uncheck both Desktop and Standard to get a bare minimum install.

Now we can install general software. This is where the dpkg
package system really shines and the "apt-get" front end is awesome.

Before we do the next step, you will need to edit the sources.list file:

```
# vi /etc/apt/sources.list
```

Uncomment the line beginning with: deb cdrom
This will configure the dpkg package system to NOT look for packages on the cdrom.
Then we want to update our sources:

```
# apt-get update
```

Now from the console, type the following:

```
# apt-get install ssh
```

This will automatically install and start the ssh server so then
you can connect remotely in your favorite terminal program.

Then we can install all the necessary packages for this guide:

```
# apt-get install apache-ssl apache-common libapache-mod-php4 \
mysql-server mysql-common mysql-client php4-mysql \
libnet1 libnet1-dev libpcre3 libpcre3-dev autoconf automake1.9 \
libpcap0.8 libpcap0.8-dev libmysqlclient15-dev \
php4-gd php4-pear libphp-adodb vim gcc make \
php4-cli libtool libssl-dev gcc-4.1 g++
```

The above command is all one line and you should be able to just
copy and paste into your terminal which will then start downloading
all the software packages and then install them.

2. Install and configure an iptables based firewall

For this next step, you can use your favorite firewall application
to setup an iptables based firewall. Shorewall is a good choice.
For simplicity sake, we will use lokkit.

apt-get install lokkit

Now we will run the lokkit command to bring up the
simple curses based configuration screen:

lokkit

Now choose “medium” and “customize” and check off your trusted device,
allow incoming on ssh (port 22) and apache-ssl (port 443).

3. Install Snort, add the snort user & group, & install VRT rules

This step involves installing snort, adding the snort user & group,
then installing the VRT intrusion detection rules

You can either proceed with Snort 2.6 or Snort 2.4.5

**********Snort 2.6 **********
cd /usr/local/src
wget http://snort.org/dl/current/snort-2.6.0.tar.gz
tar xvzf snort-2.6.0.tar.gz
cd snort-2.6.0 (take the time to read the doc/INSTALL file)
./configure --with-mysql --enable-dynamicplugin
make
make install
****

********** Snort 2.4.5**********
# cd /usr/local/src
# wget http://www.snort.org/dl/current/snort-2.4.5.tar.gz
# tar xvzf snort-2.4.5.tar.gz
# cd snort-2.4.5 (take the time to read the doc/INSTALL file)
# ./configure --with-mysql
# make
# make install
****

Next, run the following commands:

mkdir /etc/snort
mkdir /var/log/snort
groupadd snort
useradd -g snort snort
chown snort:snort /var/log/snort

A quick summary of the VRT rules from http://snort.org/rules :

Sourcefire VRT Certified Rules are the official rules of snort.org.
Each rule has been rigorously tested against the same standards
the VRT uses for Sourcefire customers. These rules are distributed
under the new VRT Certified Rules License Agreement that restricts
commercial redistribution. There are three ways to obtain these rules:

* Subscribers receive real-time rules updates as they are available
* Registered users can access rule updates 5 days after release to subscription users.
* Unregistered users receive a static ruleset at the time of each major Snort Release

Community Rules

In addition, the VRT is pleased to announce that will be maintaining a community
ruleset that contains rules submitted by members of the open source community.
While these rules are available as is, the VRT performs basic tests to ensure that
new rules will not break Snort. These rules are distributed under the GPL and are
freely available to all open source Snort users.

Get an account at snort.org and get the "registered-user" rules.
snortrules-snapshot-CURRENT.tar.gz or snortrules-snapshot-2.4.tar.gz
Here I scp'ed the files into the /root directory on the box.

```
# cd /etc/snort
# mv /root/snortrules-snapshot-2.4.tar.gz .
# tar xvzf snortrules-snapshot-2.4.tar.gz
# cp /usr/local/src/snort-2.4.5/etc/*.conf* .
# cp /usr/local/src/snort-2.4.5/etc/*.map .
```

4. Configure and start the Snort program

Now it is time to configure and start snort. First, we need to edit the main configuration file:

```
# vim /etc/snort/snort.conf
```

To get started I just changed these lines:
var RULE_PATH /etc/snort/rules
var HOME_NET 192.168.1.0/24
var EXTERNAL_NET !$HOME_NET
These variables are important in terms of "tuning" your sensor.
One must customize the pre-defined variables to make sure that
the system evaluates relevant network traffic. You want your
snort box to produce good data that you want to analyze.

Now we can make a very simple local rule in order to get
snort alerts very quickly when we start snort for the first time:
vim /etc/snort/rules/local.rules

Make a simple rule like this for testing:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; \
dsize:8; itype:8; sid:10000001;)
```

Or to REALLY generate some alerts, make this local rule:

```
alert tcp any any -> any any (msg:"test"; sid:1000002;)
```

Now we can start snort:

```
# /usr/local/bin/snort -Dq -u snort -g snort -c /etc/snort/snort.conf
```

So snort should initialize successfully. Look at /var/log/syslog for
a line that looks like this:

```
snort[1731]: Snort initialization completed successfully (pid=1731)
```

Also, /var/log/messages will show eth0 entering promiscuous mode. If you have eth1, then you can plug that interface into a tap, or into your main switch with the configured "span" mirrored port to sense and watch all the traffic on the network.

Now bring up your "sniffing" interface:

```
# ifconfig eth1 up
```

Then use the snort startup commands above along with the -i option to call an interface you would like to sniff. (i.e. -i eth1)
Remember to kill the existing snort process first before you start a new one.

A great way to sniff live traffic is take an old 10MB 4 port dumb hub and plug your ISP connection into the hub. Then you can plug your router/gateway connection into the hub. The third connection into the hub will be your sensing interface (i.e. eth1) from the snort box. There will be a total of 3 connections on this dumb hub. Now all the traffic in and out of your network is broadcasted on the ports of the dumb hub and the traffic is able to be "sniffed" by the snort box.

5. Setup the MySQL server

The next step is to setup the MySQL server. Simply doing and "apt-get install" of programs such as apache and mysql on Debian, the programs are installed and started along with the init scripts configured for startup at boot time.
First set the mysql root password by doing this command:

```
# mysqladmin -u root password "mypassword"
```

Then get into the mysql command prompt:

```
# mysql -u root -p  (then enter your password "mypassword" to get the prompt)
```

Create the snort database:

```
mysql> create database snort;
```

Create the snort user and privileges:

```
mysql> grant CREATE, INSERT, SELECT, DELETE, UPDATE \
on snort.* to snort@localhost;
```

Set the snort user password for the database:

```
mysql> SET PASSWORD FOR snort@localhost=PASSWORD('mypassword');

mysql> exit
```

Now we have to import the schema that comes with the snort program:

```
# cd /usr/local/src/snort-x.x.x/schemas/

# mysql -u root -p < create_mysql snort
```

Now you can login to the mysql server and look at the tables created:

```
# mysql -u root -p  (enter your password again)

mysql> use snort;
mysql> show tables;
```

You should see the list of new tables you just imported.

6. Configure snort to log into MySQL and test

Now we will configure snort to log into MySQL and test the system.
Lets get snort logging alerts into the mysql database by configuring
the output plugin for database logging:

```
# vim /etc/snort/snort.conf
```

Find this line below, uncomment the line, and then add your appropriate values:

```
output database: log, mysql, user=snort password=mypass dbname=snort host=localhost
```

Go restart snort and verify its writing to the database. Easiest way is to
get into mysql and "select * from event" and you should see lots of events
if you still have the alerting going on for each packet or the icmp rule.
Or you can run this command:

```
# mysql -uroot -pmypassword -D snort -e "select count(*) from event"
```

7. Apache-SSL web server install

Now we can configure the Apache-SSL web server.

Apache-SSL is trivial with Debian. After doing the apt-get install above, Apache-SSL is installed and running. The SSLCertificateFile is created for you at install time. DocumentRoot is /var/www.

Edit the apache-ssl configuration file:

vim /etc/apache-ssl/httpd.conf

Uncomment these 2 lines:

AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps

Also, we must now enable extension=mysql.so in /etc/php4/apache/php.ini

vim /etc/php4/apache/php.ini

uncomment this line:

extension=mysql.so

Then restart apache:
/etc/init.d/apache-ssl restart

8. Install and configure Basic Analysis and Security Engine (BASE)

Base is an excellent application that provides a web front-end to query and analyze the alerts coming from the snort program.
BASE 1.2.5 (sarah) was just released. From the email announcement:
This release comes after two months of enormous amounts of effort. The team, and users have fixed more bugs and implemented more features then any of our other releases to date! The CHANGELOG has 43 entries for this release alone.
http://sourceforge.net/projects/secureideas

Here is how we set up BASE:

cd /var/www
rm index.html
wget http://internap.dl.sourceforge.net/sourceforge/secureideas/base-1.2.5.tar.gz
tar xvzf base-1.2.5.tar.gz
mv base-1.2.5 base
chmod 777 base (just for now)

Open a browser and go to: https://192.168.1.13/base
(or whatever your IP is) (also, remember this is ssl only)

Click next, choose English, enter the path to adodb:
/usr/share/php/adodb

Database Name: snort
Database Host: localhost
Database Port: Leave blank for default! blank
Database User Name: snort
Database Password: mypass

Put in values for the authentication system then click submit.

Click "create baseag" which will:

Adds tables to extend the Snort DB to support the BASE functionality

Continue to step 5 to login.

You should be all setup now. I see thousands of events from my very noisy rule. Now I will disable the rule, restart snort, delete all these events from Base, and carry of with tuning my system.

Go back and chmod 755 the base directory in /var/www

With the current state of Debian Testing you must do this to get graphing to work:

First you need to link php on Debian to php4 so do this:

```
# rm /etc/alternatives/php
# ln -s /usr/bin/php4 /etc/alternatives/php
```

Then run this command:

```
# pear config-set preferred_state alpha
```

Then you have to uncomment extension=gd.so in /etc/php4/cli/php.ini since pear command line use php-cli to check dependencies

```
# vim /etc/php4/cli/php.ini
```

Uncomment this line:
extension=gd.so

Then run these commands:

```
# pear install Image_Color
# pear install Image_Canvas
# pear install Image_Graph
```

Restart apache-ssl before you click on the graphing link:

```
# /etc/init.d/apache-ssl restart
```

9. Install Barnyard and configure snort for fast unified logging

Barnyard is a program developed to perform event processing from snort's "unified file format". When one configures snort for unified logging, the snort engine can now become more efficient and focus on capturing and analyzing packets. Snort no longer has to use resources on the output plugins to inject the event data into the MySQL database as Barnyard now takes over that responsibility.

Run the following commands:

```
# cd /usr/local/src
# wget http://www.snort.org/dl/barnyard/barnyard-0.2.0.tar.gz
# tar xvzf barnyard-0.2.0.tar.gz
# cd barnyard-0.2.0

# ./configure --enable-mysql
# make
# make install
# cp /usr/local/src/barnyard-0.2.0/etc/barnyard.conf /etc/snort
```

Lets modify snort.conf first to make required changes:

```
# vim /etc/snort/snort.conf
```

Comment out this line:

output database: log, mysql, user=snort password=password dbname=snort host=localhost

so it looks like this:

#output database: log, mysql, user=snort password=password dbname=snort host=localhost

Then uncomment these 2 lines:

```
# output alert_unified: filename snort.alert, limit 128
# output log_unified: filename snort.log, limit 128
```

so they look like this:

```
output alert_unified: filename snort.alert, limit 128
output log_unified: filename snort.log, limit 128
```

Kill the existing snort process and start a new one for these changes to take affect.

Now lets configure barnyard:

```
# vim /etc/snort/barnyard.conf
```

Configure the hostname:

config hostname: testbla

Configure the listening interface:

config interface: eth1

Configure the output. Disable all of the output plugins and enable this one:

output log_acid_db: mysql, database snort, server localhost, user snort, password password, detail full

Now we want to create the waldo file so we can use Barnyard's continuous with
checkpoint mode of operation:

```
# cd /etc/snort
# vi bylog.waldo
```

Enter the following into the bylog.waldo file:

```
----------------
/var/log/snort
snort.log
108247783
0
---------------
```

The third line in the bylog.waldo file is the timestamp associated with
the particular snort unified log file in /var/log/snort. After you start
snort with the new changes, you can enter your new timestamp
number into line 3 of your bylog.waldo file. Save the file and quit.

Start Barnyard with this command:

```
# /usr/local/bin/barnyard -c /etc/snort/barnyard.conf -g \
/etc/snort/gen-msg.map -s /etc/snort/sid-msg.map -d /var/log/snort -f \
snort.log -w /etc/snort/bylog.waldo &
```

Now you can run this command to make sure that barnyard is correctly
inserting events into the mysql database:

```
# mysql -uroot -pmypassword -D snort -e "select count(*) from event"
```

10. Keep your rules up to date with Oinkmaster.

If you have many sensors, it can be a very difficult job to keep all the
rules current on all of your sensors. A great tool for this task is Oinkmaster
and the new 2.0 version was just released in February: http://oinkmaster.sourceforge.net/

Oinkmaster can do many things to help you automate rule management such as:
- Download updates from a main site or multiple locations at the same time

- Add new rules to your installation and preserve rules that have been commented
- Can merge new variables from snort.conf in the distribution tarball into your local copy.
- Can backup your old rules before overwriting them with the new ones.
- much more

Run the following commands to do the basic install:

```
# cd /usr/local/src
# wget http://internap.dl.sourceforge.net/sourceforge/oinkmaster/oinkmaster-2.0.tar.gz
(or whatever mirror you choose to download from)
# tar xvzf oinkmaster-2.0.tar.gz
# cd oinkmaster-2.0
# cp oinkmaster.pl /usr/local/bin
# mkdir /usr/local/etc
# cp oinkmaster.conf /usr/local/etc
```

Now let's work on the configuration file. There are many options lots of different things one can do with oinkmaster, but for now, the objective is to simply download a rule pack from snort.org.

```
# vim /usr/local/etc/oinkmaster.conf
```

Note this section of the configuration file:

```
---------------------------------------------------------------------------------------------
# As of March 2005, you must register on the Snort site to get access
# to the official Snort rules. This will get you an "oinkcode".
# You then specify the URL as
# http://www.snort.org/pub-bin/oinkmaster.cgi/<oinkcode>/<filename>
# For example, if your code is 5a081649c06a277e1022e1284b and
# you use Snort 2.4, the url to use would be (without the wrap):
# http://www.snort.org/pub-bin/oinkmaster.cgi/
# 5a081649c06a277e1022e1284bdc8fabda70e2a4/snortrules-snapshot-2.4.tar.gz
# See the Oinkmaster FAQ Q1 and http://www.snort.org/rules/ for
# more information.
# URL examples follows. Replace <oinkcode> with the code you get on the
# Snort site in your registered user profile.
# Example for Snort 2.4
# url = http://www.snort.org/pub-bin/oinkmaster.cgi/<oinkcode>/snortrules-snapshot-2.4.tar.gz
---------------------------------------------------------------------------------------------
```

Once you get your oinkcode you will be able to automate your rule management. Oinkmaster takes the output directory as an argument with the -o switch. This tells Oinkmaster where you want the downloaded rules unpacked. This is most likely where your production snort rules are located. Your rules will get replaced per the way the oinkmaster.conf file was configured. Try these commands for a first try:

```
# mkdir /tmp/oinktest
# /usr/local/bin/oinkmaster.pl -o /tmp/oinktest
```

Take a look in /tmp/oinktest for a pleasant surprise:

```
# ls -al /tmp/oinktest/
```

Using the -b switch will make Oinkmaster back up the current rules in the location you specify before doing the actual update. We now have a copy of the rules on /tmp/oinktest, so let's try backing them up first before we download again. First we must modify one of the existing rules like this:

vim /tmp/oinktest/pop3.rules (replace the port number 110 with something random)

Then run these commands:

mkdir /tmp/OINKBACK

/usr/local/bin/oinkmaster.pl -o /tmp/oinktest -b /tmp/OINKBACK

You will see the output stating "Modified Active Rules" and then you will have a backup in the /tmp/OINKBACK directory like this:

-rw-r--r-- 1 root root 241K 2006-03-13 17:54 rules-backup-20060313-175427.tar.gz

11. Startup script for snort & barnyard

Create the startup script:

#vim /etc/init.d/snort-barn

Enter the following into the file:

```
---------------------------------------------------------------------------
#!/bin/bash
/sbin/ifconfig eth1 up
/usr/local/bin/snort -Dq -u snort -g snort -c \
/etc/snort/snort.conf -i eth1
/usr/local/bin/barnyard -c /etc/snort/barnyard.conf -g \
/etc/snort/gen-msg.map -s /etc/snort/sid-msg.map -d \
/var/log/snort -f snort.log -w /etc/snort/bylog.waldo &
---------------------------------------------------------------------------
```

Make it executable:
#chmod +x /etc/init.d/snort-barn

The command `update-rc.d` will set up links between files in the directories rc?.d
#update-rc.d snort-barn defaults 95

Reboot and see if it works!

Conclusion:

In conclusion, network security is a serious matter that deserves special attention. Using a powerful commodity OS like Debian along with exciting software such as Snort, Base, Barnyard, and Oinkmaster, one is able to build a homebrewed IDS sensor which allows one to monitor network segments for intrusion activity.