5 Test Implementation

The test implementation phase mainly involves configuring Snort and conducting the test runs. A description of the standard configuration used is provided in section 5.1. Section 5.1.1 describes how the ruleset section of the configuration file can be manipulated to create different configurations. Then the resulting configurations of Snort created for the testing are discussed in section 5.2. After that, an example of a test run is presented.

5.1 Configuring Snort

Snort needs to be configured in order to get it up and running. This configuration is done in the main configuration file (snort.conf). A sample Snort configuration file that was used in the Snort 1.7 Full test run is included in Appendix J. Generally, the following steps need to be taken to configure Snort:

1. Set the network variables to reflect the network environment
2. Configure the preprocessors
3. Configure output plugins
4. Customize the ruleset

In step 1, various variables like HOME_NET, EXTERNAL_NET and DNS_SERVERS need to be defined to reflect the network topology. If the wrong addresses are entered, Snort may output false or no alarms. The HOME_NET variable indicates the home address range and the EXTERNAL_NET variable indicates the external network address space used. The DNS_SERVER variable indicates the domain name servers. These variables were defined as follows (see Appendix J):

```
var HOME_NET $eth0_ADDRESS
var EXTERNAL_NET !$HOME_NET
var DNS_SERVERS $HOME_NET
```

In step 2, the preprocessors need to be configured. Preprocessors, such as *frag2* and the *stream4* module, are plugins that fit into the detection scheme before the detection engine does its work. *Frag2* is an IP defragmentation processor. The *stream4* module provides TCP stream reassembly and stateful analysis capabilities for Snort. This module consists of two preprocessors called *stream4* and *stream4_reassemble* that must be activated. Two other preprocessors used are *portscan* and *portscan-ignorehosts*. *Portscan* is responsible for detecting portscans and *portscan-ignorehosts* is responsible for ignoring portscan detection from the specified host(s). The preprocessors used in the test runs were defined as follows (see Appendix J):

```
preprocessor frag2
preprocessor stream4
preprocessor stream4_reassemble
preprocessor portscan: $HOME_NET 5 3 $LOGDIR/$INSTANCE.scan
preprocessor portscan-ignorehosts: 0.0.0.0
```

In step 3, the output plugins need to be configured. The output plugins work with the alert messages. By default, all logs are written in the /var/log/snort directory, and all of the alerts are written to the /var/log/snort/alert file.

In step 4, the ruleset need to be specified. The rules are a vital part of Snort. There are various categories of rules that come with Snort installations. They can be found in the /etc/snort directory, ending with *.rules. The rules specified in the Snort 1.7 configuration were as follows:

```
include local.rules
include exploit.rules
include scan.rules
include finger.rules
include ftp.rules
include telnet.rules
include smtp.rules
include rpc.rules
include rservices.rules
include backdoor.rules
include dos.rules
include ddos.rules
include dns.rules
include netbios.rules
include sql.rules
include web-cgi.rules
include web-coldfusion.rules
include web-frontpage.rules
include web-misc.rules
include web-iis.rules
include icmp.rules
include misc.rules
include policy.rules
include info.rules
include virus.rules
```

The network variables, preprocessors, and output plugins used in the test runs were the standard configuration settings that came with the Snort installation. The portion of the configuration file that was customized was the ruleset section which is discussed in section 5.1.1.

5.1.1 Specifying the Snort Rulesets

The four Snort configurations created were differentiated by the ruleset specified for that particular configuration. Rulesets are specified in the ruleset section of the main configuration file which is step 4 in configuring Snort. The following is an example of a full ruleset that came with the Snort 1.7 initial bundle (comments are preceded with the '#' symbol):

```
#--------------------------------------------------
# http://www.snort.org Snort 1.7 Ruleset
# Contact: snort-sigs@lists.sourceforge.net
#--------------------------------------------------
# NOTE:This ruleset only works for 1.7.0 and later.
#--------------------------------------------------
include local.rules
```

```
include exploit.rules
include scan.rules
include finger.rules
include ftp.rules
include telnet.rules
include smtp.rules
include rpc.rules
include rservices.rules
include backdoor.rules
include dos.rules
include ddos.rules
include dns.rules
include netbios.rules
include sql.rules
include web-cgi.rules
include web-coldfusion.rules
include web-frontpage.rules
include web- misc.rules
include web- iis.rules
include icmp.rules
include misc.rules
include policy.rules
include info.rules
include virus.rules
```

The configuration is altered if any of the above rules are excluded or new rules added. An example of how some of the rules are excluded, hence differentiating the configuration, is shown below:

```
#---------------------------------------------------
# http://www.snort.org Snort 2.7 Ruleset
# Contact: snort-sigs@lists.sourceforge.net
#---------------------------------------------------
# NOTE:This ruleset only works for 2.7.0 and later
#---------------------------------------------------
#include local.rules
#include exploit.rules
include scan.rules
include finger.rules
include ftp.rules
include telnet.rules
include smtp.rules
include rpc.rules
include rservices.rules
include backdoor.rules
include dos.rules
include ddos.rules
include dns.rules
include netbios.rules
include sql.rules
include web-cgi.rules
include web-coldfusion.rules
include web-frontpage.rules
include web- misc.rules
include web- iis.rules
include icmp.rules
include misc.rules
```

```
include policy.rules
include info.rules
include virus.rules
```

The above example illustrates the exclusion of the 'local' and 'exploit' rules by commenting off the `include local.rules` and `include exploit.rules` lines.

The initial installations of Snort 1.7 and Snort 1.8.3 used in this research come with full rulesets. The rulesets were modified using the abovementioned method, that is. by commenting off certain rules, in order to create the four different configurations described in the following section.

5.2 Resulting Configurations

Snort 1.8.3 was run with two different configurations. One was run with a complete set of rules minus the icmp-info rules. This configuration is termed here as Snort 1.8 Full. The second configuration of Snort 1.8.3 had more than just the icmp-info rules removed. This configuration is termed as Snort 1.8 Custom.

Snort 1.7 was also run with two different configurations. One was run with a complete set of rules and is termed as Snort 1.7 Full. The other configuration, termed as Snort 1.7 Custom, was activated with a few rules removed.

The ruleset for each of the Snort configurations described above are included in Appendix E of this dissertation.

5.3 Testing the 4 Configurations of Snort

After the configurations were completed, each of the four test data files were ran through the four configurations of Snort which resulted in sixteen test runs.

The alerts that were generated were noticeably large. Anywhere from zero to thousands of alerts had been generated, depending on the log file inputted. The task of going through these text files manually was going to be too time and labour intensive. Snort lists the number of packets scanned and alerts found when it has completed running. In order to eliminate the overwhelming manual task, SnortSnarf was used to process the massive alert files and put them into easy-to-read HTML format.

5.4 Example of a Test Run

In the /root directory, the command snort –c snort.conf –r data-set-1 was entered. This command tells Snort to read the test data file (data-set-1) and raises alerts according to the ruleset defined in the configuration file, snort.conf. The typical output from the start-up of Snort can be seen in Figure 5.1.

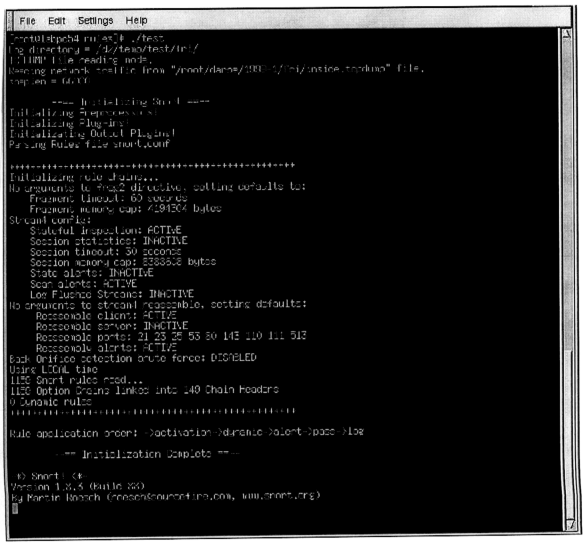

Figure 5.1: Snort Pre-Run Screen

Once the test run is completed, that is, when Snort completes reading the test data file, the following end-of-run screen (Figure 5.2) is displayed:

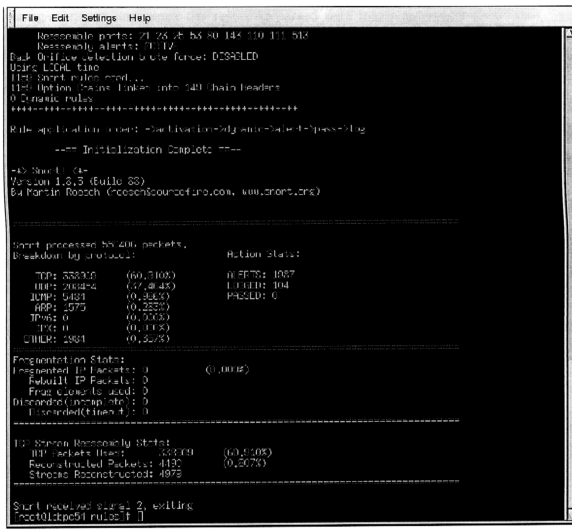

Figure 5.2: Snort Post-Run Screen

The Snort run creates an "alerts" file. This resultant alert text file is now ready to be parsed into HTML using SnortSnarf. This was executed by entering the following command:

```
perl /etc/SnortSnarf/snortsnarf.pl –d /var/www/html/snortsnarf /var/log/snort/alert
```

The command perl /etc/SnortSnarf/snortsnarf.pl runs Snortsnarf. The command –d /var/www/html/snortsnarf /var/log/snort/alert tells it to process the Snort alert file from

/var/log/snort/alert and place the resulting parsed data into /var/www/html/snortsnarf. The parsed data is then formed into easy-to-read HTML format. This was a much better method to view the data, because it links all the data and sorts it by alert type and IP.

A screen shot of the typical HTML output can be seen in Figure 5.3.

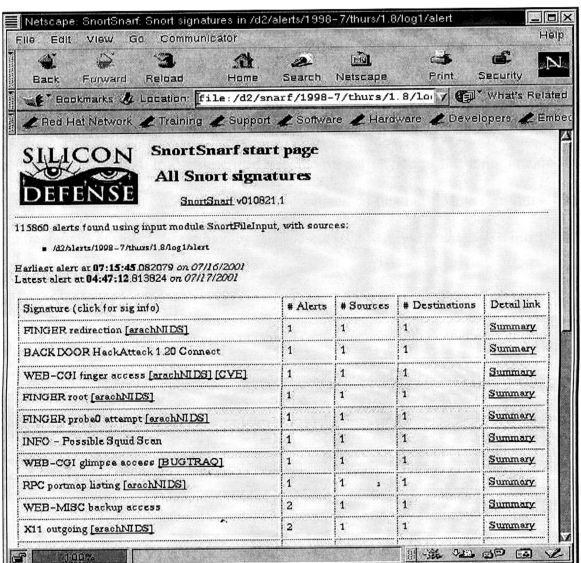

Figure 5.3: SnortSnarf Output

After the SnortSnarf run is completed, the results are transferred into their corresponding results forms as described in section 4.3.1 under subsection (F) on Result Forms.

5.5 Summary

This chapter is an overview of the test implementation phase. To begin with, the first section described the standard configuration setting used in all the four Snort configurations created. It also described how the ruleset section of the configuration file can be manipulated to create different configurations. Following this, the resulting four configurations of Snort, determined by the rulesets specified, were addressed. Finally, an example of a test run is presented.