# Department of Computing

**CS381: Network Security**

**Class: BSCS 4 (G1)**

# Lab 11: Intrusion Detection System

**CLO4: Compare, investigate and evaluate different security mechanisms to protect network resources**

**Date: December 18, 2017**

**Time: 9 am- 12 pm**

# Instructor: Ms. Haleemah Zia

# Lab 11: Intrusion Detection System

**Introduction**

This lab is based on introduction to intrusion detection system, using snort. Snort is an open source intrusion detection/prevention system created by Martin "Marty" Roesch, founder of Source fire. It is capable of performing real-time traffic analysis and logging. It is the most widely used IDS/IPS system. It can monitor for, detect and respond to various attack strategies by using signature, protocol and anomaly-based inspection techniques.

Many security tools run primarily on Linux/Unix (\*nix) platforms. For various reasons, many system administrators have been raised on Windows Operating Systems. This lack of security tool availability can be frustrating to these system administrators. Luckily, Snort functionality has been ported to Windows.

For an average Windows user, installing Snort is a little more of a headache than for your average Linux user. This is because Snort was developed initially for open-source Unix-like platforms such as Linux, and if you are at all familiar with Linux, you know what that means: command-line options and text-based configuration files. For a Windows user who’s used to point-and-click configuration, command-line is a little intimidating. Add to that the fact that there’s little supporting documentation for the Windows platform on Snort’s Web site or the rest of the Internet, and you have all the makings of a bumpy ride. To perform a basic install of Snort on a Windows system, we need to download a couple of programs. First, download the Snort Installer binary from [www.snort.org](http://www.snort.org)

**Objectives**

Students will learn about IDS and the following features:

1. Protocol analysis and content searching/matching.
2. Detecting a variety of attacks and probes, such as buffer overflows, stealth port scans
3. Use of flexible rules languages to describe traffic that it should collect or pass, as well as detection engines that utilize a modular plug-in architecture.
4. Real-time alerting capability

**Tools/Software Requirement**

SNORT, WinPCap, Windows/Linux/OSX

**Description**

For Snort to work properly, we will need to put our network interface card (NIC) into promiscuous mode where it can see all traffic the flows to it. To do this on Windows, we will need to download the Windows Packet Capture library from <http://www.winpcap.org/>

**Installation For Windows**

After you have downloaded the files referenced above, installation is pretty straightforward. If you have User Account Control (UAC) enabled, you will need to answer "Yes" when prompted "Do you want to allow the following program to make changes to this computer?" for both WinPcap and Snort.

**Installing WinPcap**

First we will start by installing the WinPcap libraries so we can sniff all the packets from our NIC. Installation of WinPcap is pretty easy. WinPcap (Windows Packet Capture Library) is a packet-capture driver. Functionally, this means that WinPcap grabs packets from the network wire and pitches them to Snort.

The WinPcap driver performs these functions for Snort:

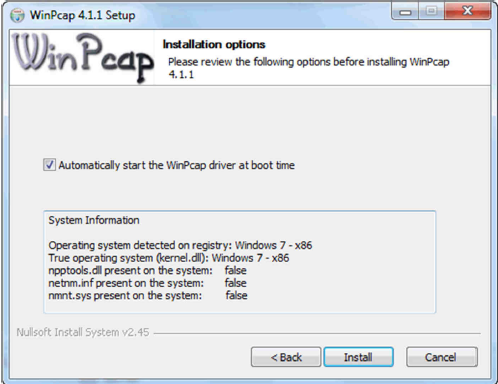
1. Obtains a list of operational network adapters and retrieves information about the adapters.
2. Sniffs packets using one of the adapters that you select.
3. Saves packets to the hard drive (or more importantly for us, pitches them to Snort).

WinPcap is a Windows version of libpcap, which is used for running Snort with Linux. For our installation we will be accepting all of the default settings. To start the installation, navigate to the location of the WinPCap file we have downloaded.

1. Right click the file and select "Run as Administrator". You will be presented a title screen.



1. Just click "Next" to continue to the Welcome screen.
2. Click "Next" again to continue to the License Agreement screen.
3. To continue installation of WinPCap, you must agree to the license terms by clicking "I Agree". If you do not agree to the terms of the license agreement, you will not be able to install WinPCap. Once you click "I Agree", you can continue the installation with Installation Options.
4. We will leave the following settings as they are and click "Install". This will start the installation process. This process will not take very long and you will see the Completion screen next.
5. Click "Finish" to exit the WinPcap setup application. You will need to reboot your computer at this time.

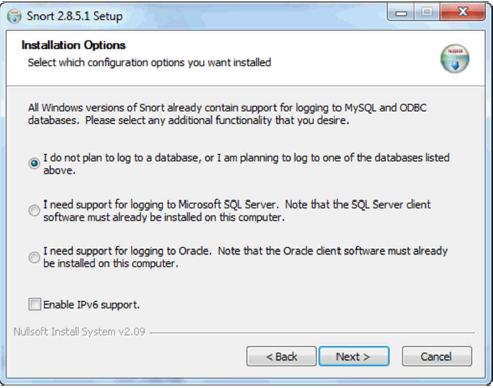


1. Click "Finish" to exit the WinPCap setup application. You will need to reboot your computer at this time.

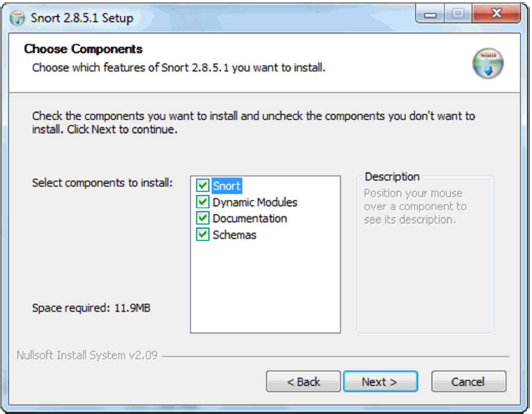
**Installing SNORT**

Now we are ready to move on and install Snort.

1. To install Snort, navigate to the location of the Snort Installer file. Right click the file and select "Run as Administrator". As we did with WinPcap, we will also be installing Snort with all the default settings. The setup application will launch and prompt you to read and agree to the License Agreement.
2. Click "I Agree" if you are satisfied with the License Agreement. If you do not agree, you will not be able to install Snort. Once you click "I Agree", setup will continue with the next step and prompt you for the installation type.



1. Select the first radio button as seen above and click "Next" to select the Snort components to install.



1. Let's leave all the components selected and click "Next" to move on to select our installation location.
2. For our install, let's take the default location of "c:\snort" and click "Next". At this point we have given the setup application all of the information it needs to extract the files necessary for our installation.
3. Once the files are extracted, we will need to click "Close" to exit the setup application. The setup application will alert you to make sure a minimum version of WinPcap is installed (which we have completed) and that we need to edit the Snort configuration file.
4. Click "OK" to acknowledge this and close the setup application. You may get a notification from the Windows Program Compatibility Assistant warning that the program might not have installed correctly. If you do, select "This program installed correctly" and continue on.

**Instructions**

Complete the tasks below and insert the solution/answer in this document as directed below. You must show the execution of below tasks (by adding the snapshots), along with your required commands to get your work graded. You must also submit the completed Word document on LMS before the deadline. You can get help from the Internet, but copying is not allowed.

**Lab Tasks**

**Configuration On Windows**

A new Snort installation requires a few configuration points. Conveniently, one file has all the configuration settings required C:\etc\snort.conf

Double-check everything you type in to the snort.conf file. If entries aren’t exactly correct, Snort doesn’t work.

**Network settings**

The network settings allow you to set Snort to monitor any range of network IP addresses, from a single IP address, several IP addresses in groups or individually, and entire IP subnets. You can configure the IP address range and the subnet. You can control the network range that Snort monitors by changing the Var HOME\_NET setting in snort.conf. Your options are:

1. Entire network. By default, snort.conf contains the following line, which monitors the entire local network: **var HOME\_NET** any If you don’t change this setting, Snort monitors the entire network segment the Snort system is attached to.
2. Single IP address: To monitor a single IP or computer insert the IP address range and the subnet of the network or host into snort.conf. To do this, replace the existing **var HOME\_NET** configuration line with this form: **var HOME\_NET IP Address Range/Subnet**
3. The IP Address Range/Subnet notation uses CIDR notation. For example this monitors the entire Class C network: **var HOME\_NET 192.168.10.0/24**. And this line monitors a single host on the Class C network: **var HOME\_NET 192.168.10.2/32**
4. Multiple hosts: You can specify a number of hosts within the network space you are monitoring by listing them in the var HOME\_NET configuration statement. The line takes this form: **var HOME\_NET IP Address Range/Subnet,IP Address Range/Subnet,...**
   1. Separate each IP address in the **var HOME\_NET** configuration statement by a comma without spaces. If there are any blank spaces in the list of IP addresses and subnets, then Snort fails to start. The following example monitors three hosts on a typical class C network: **var HOME\_NET 192.168.10.2/32,192.168.10.3/32,192.168.10.6/32**

**Rules**

For Snort to detect attacks and alert you when attacks occur, Snort needs to know where its rule base is (and you need to know it if you want to write new rules). By default, the rulebase is in

C:\Snort\rules

To set the rules path in the snort.conf file, replace the existing **var RULE\_PATH** line.

**Output settings**

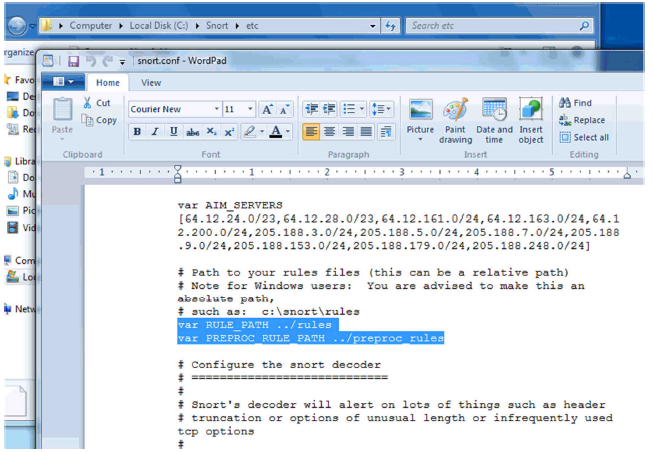
Output settings are very important in Snort, for they define how Snort’s information will be presented to you. We are concerned with configuring Snort to output to an alert text file.

**Include configuration**

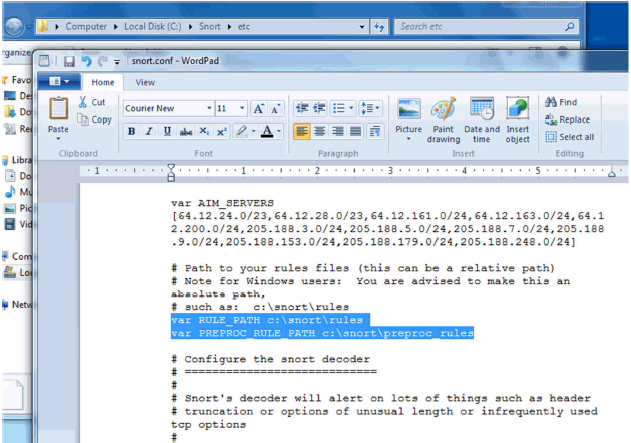
Two standard Snort configuration files must be referenced for Snort to properly classify and provide references to the alerts it generates: **classification.config** and **reference.config**. **classification.config** holds alert levels for the rules that Snort monitors against network traffic. **reference.config** contains URLs referenced in the rules that provide more information about the alert event.

**Set Snort Path**

As we were told by the Snort setup application, we will need to change a couple of parameters in the c:\snort\etc\snort.conf file. To do so, let's use Microsoft's Wordpad application. Open the snort.conf file and find the lines highlighted below:



Once you find these lines, modify them to reflect our default install path (c:\snort) as seen below:



**Fix VAR**

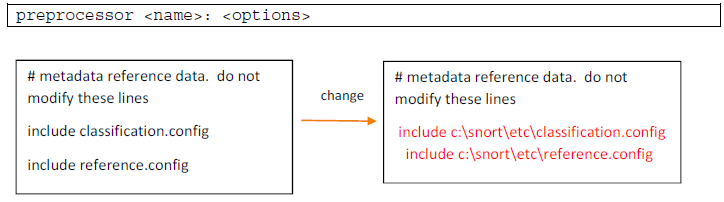
Here error is in **line no. 45** go to the line no 45 and replace word:

**"Ipvar to var**" (replace all)

**Fix Path for Dynamic Preprocessor**

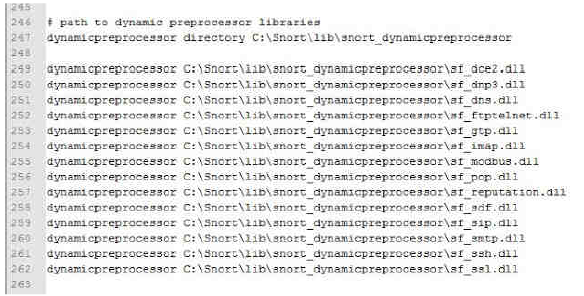
Preprocessors were introduced in version 1.5 of Snort. They allow the functionality of Snort to be extended by allowing users and programmers to drop modular plugins into Snort fairly easily.

Preprocessor code is run before the detection engine is called, but after the packet has been decoded. The packet can be modified or analyzed in an out-of-band manner using this mechanism. Preprocessors are loaded and configured using the preprocessor keyword. The format of the preprocessor directive in the Snort config file is:



Also change the path which will be like this

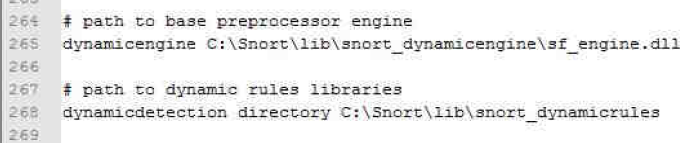
C:\snort\lib\snort\_dynamicpreprocessor\



**Fix Path for Dynamic Engine Rules**

Change the path for dynamicengine and dynamic rules With this c:\snort\lib and change the ".SO" extension to ".dll"

Which will look like this:

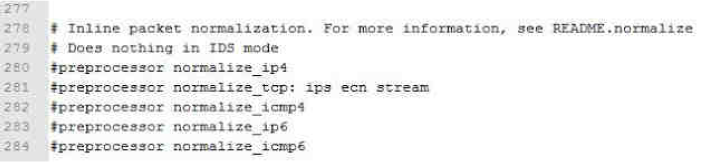


Also make a folder named **snort\_dynamicrules** in C:\snort\lib\

**Preprocessor Normalization**

For this:- comment all preprocessor normalize lines **(using #)**

Which will look like this:



**White & Black Lists**

Create text document in c:\snort\rules\ of name **“white\_list.rules”**

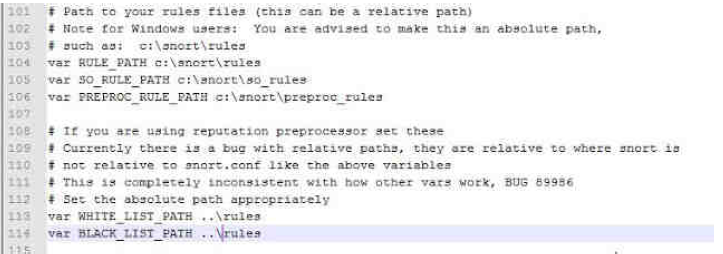
Create text document in c:\snort\rules\ of name **“Black\_list.rules”**

Now inline no. 113 and 114 that is

var WHITE\_LIST\_PATH ../rules

var BLACK\_LIST\_PATH ../rules

Change the **‘/ ’ into ‘ \ ’**



Now go to the line no. 525 and 526

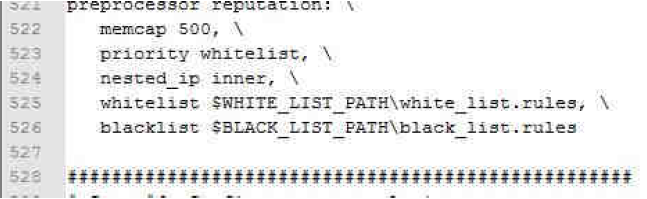
Search for these line

whitelist $WHITE\_LIST\_PATH/white\_list.rules, \

blacklist $BLACK\_LIST\_PATH/black\_list.rules

and change **‘/’ into ‘\’**

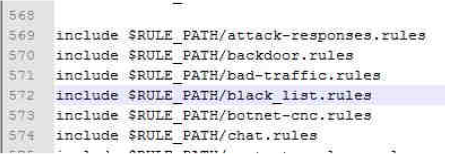
Which will look like :-



Now go to the line no. 572 which is include $RULE\_PATH/blacklist.rules

Change the name blacklist into black\_list…

Which will look like:-



**Test Your Snort Configuration**

With so many new changes you need to check if the snort.conf is correct. You can do this using:

snort -i 1 -l c:\snort\log -c c:\snort\etc\snort.conf -T

If you made all above changes correctly, it should give you a message



**Snort Resource Requirements**

All Windows-based operating systems have high base hardware requirements relative to other operating systems, even with as much unnecessary stuff removed as possible. When it comes to recommended hardware, for Snort, the faster and more the better. Snort needs as much processor speed and memory you can throw at it, relative to the activity on your network: If Snort runs out of resources, it drops packets; it won’t analyze all of the network packets that come under its nose. With Snort dropping packets, the entire purpose of IDS is defeated; an attack on your network or hosts can come at any time. (Murphy’s Law says the attack will probably come when your IDS is overloaded.)

**Snort Modes**

Snort can be configured to run in three modes:

1. Sniffer mode, which simply reads the packets off of the network and displays them for you in a continuous stream on the console (screen).
2. Packet Logger mode, which logs the packets to disk.
3. Network Intrusion Detection System (NIDS) mode, the most complex and configurable configuration, which allows Snort to analyze network traffic for matches against a user-defined rule set and performs several actions based upon what it sees

**Sniffer Mode**

1. In a command prompt, cd to c:\snort
2. .\bin\snort help >>> View different options for snort. You can also run Snort with the "-?" option to get a full list of options available
3. Use the appropriate flag to list available interfaces

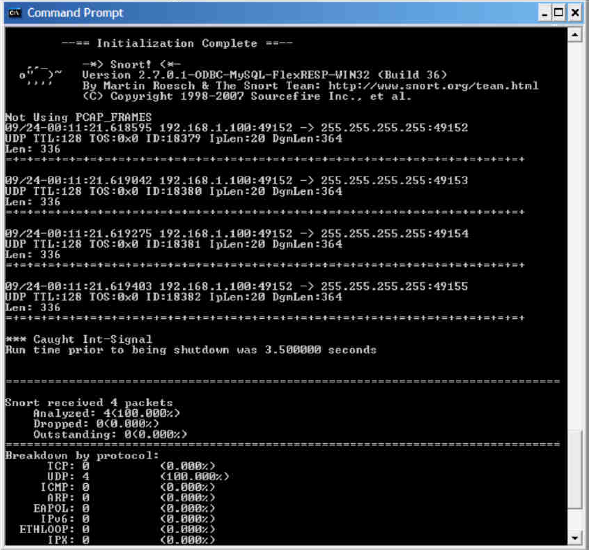
**Task 1:** What flag did you use to get the list of available interfaces? [1 Mark]

|  |
| --- |
| Available Interfaces |
| Snort –W |

1. Run snort in the sniffer mode by typing

.\bin\snort –v –i 1

**Note:** You need to replace the “i 1” with whichever your network interface is (see point 3 above). Also note that this lab assumes that you are not using a wireless interface. If you want to use a wireless NIC card, then you need to install a Pcap for wireless traffic like AirPcap. You should see something like this:



Ctrl-c will stop the capture. Notice that no data-link headers are being displayed.

**Task 2:** Find the flag that will display data-link headers? [1 Mark]

|  |
| --- |
| Data-link Header |
| [\\add](file:///\\add) the snaps here, which shows the command (you use) and the headers  Snort –e |

**Task 3:** Find the flag that will display application layer data? [1 Mark]

|  |
| --- |
| Application Layer Data |
| [\\add](file:///\\add) the snaps here, which shows the command (you use) and the application layer data  Snort –d |

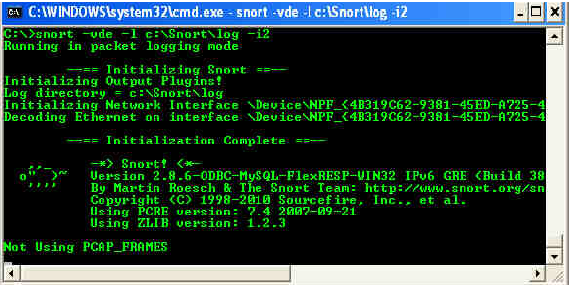
**Task 4:** How to see ARP packets? [1 Mark]

|  |
| --- |
| ARP Packets |
| [\\add](file:///\\add) the snaps here, which shows the command (you use) and the ARP packets  The command we have to use is –a to capture ARP packets . Since we are using windows , so we cant capture the ARP packets |

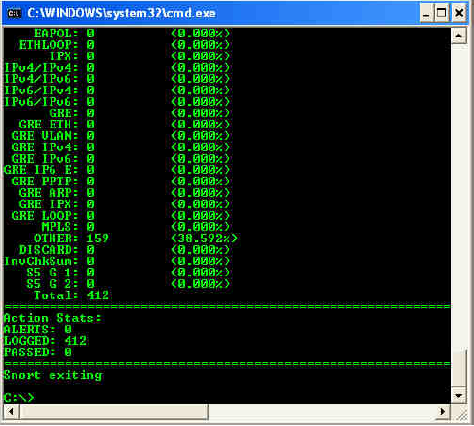
**Logger Modes**

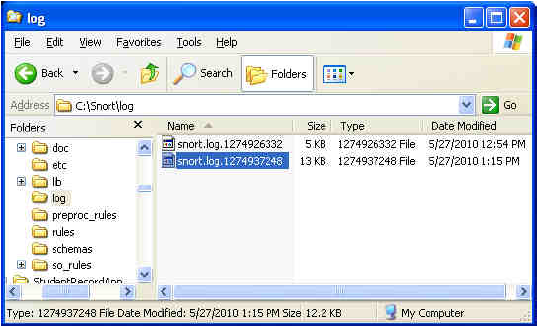
You can test Snort’s logging abilities with the -l (log) switch, by typing (take note on the order of the options):

C:\>snort -vde -l c:\Snort\log –i 1

****

Please press Ctrl+C to stop.

****

****

This runs Snort in descriptive verbose mode and logs all its findings to the directory called log under the Snort installation directory. The individual packets are filed in hierarchical directories based on the IP address from where the packet was received. Several command-line switches are specific to logging and output, including the ability to log all packets to a single binary file. Play around with those as needed.

**IDS Modes Using Snort Rules**

Snort rules operate on network (IP) layer and transport (TCP/UDP) layer protocols. However there are methods to detect anomalies in data link layer and application layer protocols. The second part of each Snort rule shows the protocol and you will learn shortly how to write these rules. A Snort rule can be broken down into two basic parts, the rule header and options for the rule. The rule header contains the action to perform, the protocol that the rule applies to, and the source and destination addresses and ports. The rule options allow you to create a descriptive message to associate with the rule, as well as check a variety of other packet attributes by making use of Snort's extensive library of plug-ins. Here's the general form of a Snort rule:

action proto src\_ip src\_port direction dst\_ip dst\_port (options)

When a packet comes in, its source and destination IP addresses and ports are then compared to the rules in the ruleset. If any of them are applicable to the packet, then the options are compared to the packet. If all of these comparisons return a match, then the specified action is taken.

Snort provides several built-in actions that you can use when crafting your rules. To simply log the packet that matches a rule, use the log action. The alert action generates an alert using the method specified in your configuration file or on the command line, in addition to logging the packet. One nice feature is that you can have very general rules and then create exceptions by writing a rule that uses the pass action. This works especially well when you are using the rules distributed with Snort, but are frequently getting false positives for some of them. If this happens and it's not a security risk to ignore them, you can simply write a **pass** rule for it.

**The First Bad Rule**

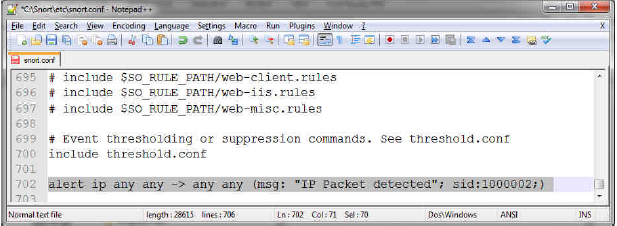
Here is the first (very) bad rule. In fact, this may be the worst rule ever written, but it does a very good job of testing if Snort is working well and is able to generate alerts.

alert ip any any -> any any (msg: "IP Packet detected";)

You can use this rule at the end of the snort.conf file the first time you install Snort. The rule will generate an alert message for ***every***captured ***IP***packet. It will soon fill up your disk space if you leave it there! This rule is bad because it does not convey ***any***information. What is the point of using a rule on a permanent basis that tells you nothing other than the fact that Snort is working? This should be your first test to make sure that Snort is installed properly. In the next section, you will find information about the different parts of a Snort rule. However for the sake of completeness, the following is a brief explanation of different words used in this rule:

1. The word “alert” shows that this rule will generate an alert message when the criteria are met for a captured packet. The criteria are defined by the words that follow.
2. The “ip” part shows that this rule will be applied on all *IP* packets.
3. The first “any” is used for source *IP* address and shows that the rule will be applied to all packets.
4. The second “any” is used for the port number. Since port numbers are irrelevant at the *IP* layer, the rule will be applied to all packets.
5. The -> sign shows the direction of the packet.
6. The third “any” is used for destination *IP* address and shows that the rule will be applied to all packets irrespective of destination *IP* address.
7. The fourth “any” is used for destination port. Again it is irrelevant because this rule is for *IP* packets and port numbers are irrelevant.
8. The last part is the rule options and contains a message that will be logged along with the alert.

Edit the snort.conf file and add this bad rule. The sid:1000002 indicates the rule ID which is required by Snort to distinguish between rules

****

**Task 5:** Now execute Snort with the new rule in effect using the following command. Examine the log folder and explain why this is a bad rule. [1 Mark]

**C:\snort\bin>snort -i 1 -l c:\snort\log -c c:\snort\etc\snort.conf -K ascii**

Where:

-c = Configure file to use (role file to use)

-l = Directory to log

-K = Logging mode [pcap (default), ascii, none ]

|  |
| --- |
| Justification |
| After examining the log folder we have come to know why this is the bad rule. It is bad because this rule take log of every packet it capture which is unnecessary . So that is why it is considered as a bad rule |

**ICMP Rule**

The next rule isn't quite as bad. It generates alerts for all captured ICMP packets. Again, this rule is useful to find out if Snort is working. Now modify the rule to generate alerts for all captured ICMP packets. To try out your new rule send an ICMP ping packet (which is basically ICMP ECHO REQUEST) to your PC address using the following command and examine the log folder:

ping 192.168.2.1

Note that 192.168.2.1 is the *IP* address of gateway/router or some other host on the same network where the Snort machine is present. This command should be executed on the machine where you installed Snort. The command can be used both on UNIX and Microsoft Windows machines.

**Task 6:** Write the new rule that you used. Do you notice any logs due to this new rule?

[1 Mark]

|  |
| --- |
| Justification |
| [\\add](file:///\\add) the snaps here, which shows the command (you use). Also, provide justification about the new rule |

**Telnet**

log tcp any any -> 192.168.1.0/24 23

This rule says to log tcp traffic coming from any IP address and any source port to this network where the destination port is 23 (telnet).

**SSH Rule**

log tcp any any -> any 22 (msg:”Geez! Someone’s trying to use SSH!”;)

Here we can see how a message can be appended to the packets that we capture. In this case we’re looking for any incoming traffic going to the standard SSH port.

**Bi-directional Traffic Rule**

Rules can also be bi-directional and look for traffic going in either direction:

log tcp any any <> any 23

This rule logs all tcp traffic where the destination port is 23 (telnet).

**Specifying Port Numbers in Rules**

Port numbers may be specified in a number of ways, including "any" ports, static port definitions, ranges, and by negation. "Any" ports is a wildcard value, meaning literally any port. Static ports are indicated by a single port number, such as 111 for portmapper, 23 for telnet, or 80 for http. Port ranges are indicated with the range operator ":". The range operator may be applied in a number of ways, such as in Example below:

log udp any any -> 192.168.1.0/24 1:1024

Log UDP traffic coming from any port and destination ports ranging from 1 to 1024.

log tcp any any -> 192.168.1.0/24 :6000

Log TCP traffic from any port going to ports less than or equal to 6000.

log tcp any :1024 -> 192.168.1.0/24 500:

Log TCP traffic from privileged ports less than or equal to 1024 going to ports greater than or equal to 500.

The negation operator, "!", may be applied against any of the other rule types (except "any", which would translate to none). For example, if for some reason you wanted to log everything except the X Windows ports, you could do something like the rule below:

log tcp any any -> 192.168.1.0/24 !6000:6010

**Understanding Standard Alert Output**

When Snort generates an alert message, it will usually look like the following:

[\*\*] [116:56:1] (snort\_decoder): T/TCP Detected [\*\*]

The first number is the Generator ID, this tells the user what component of Snort generated this alert. For a list of GIDs, please read etc/generators in the Snort source. In this case, we know that this event came from the “decode” (116) component of Snort. The second number is the Snort ID (sometimes referred to as Signature ID). For a list of preprocessor SIDs, please see etc/gen-msg.map. Rule-based SIDs are written directly into the rules with the sid option. In this case, 56 represent a T/TCP event. The third number is the revision ID. This number is primarily used when writing signatures, as each rendition of the rule should be increment this number with the rev option.

**Task 7:** Read the alert message and check it from the reference provided in the message. Explain what this rule does and what it is for. [1 Mark]

|  |
| --- |
| Justification |
|  |

Another useful option is **content**, which allows you to search a packet for a sequence of characters or hexadecimal values. If you are searching for a string, you can just put it in quotes. In addition, if you want it to do a case-insensitive search, you can add **nocase**; to the end of all your options. However, if you are looking for a sequence of hexadecimal digits, you must enclose them in **|** characters. This rule will trigger when it sees the digit 0x90:

alert tcp any any -> any any (msg:"Possible exploit"; content:"|90|";

This digit is the hexadecimal equivalent of the NOP instruction on the x86 architecture and is often seen in exploit code since it can be used to make buffer overflow exploits easier to write.

The **offset** and **depth** options can be used in conjunction with the **content** option to limit the searched portion of the data payload to a specific range of bytes.

If you wanted to limit content matches for NOP instructions to between bytes 40 and 75 of the data portion of a packet, you could modify the previously shown rule to look like this:

alert tcp any any -> any any (msg:"Possible exploit"; content:"|90|"; \ offset:40; depth:75;)

You can also match against packets that do not contain the specified sequence by prefixing it with a **!**. In addition, many shell code payloads can be very large compared to the normal amount of data carried in a packet sent to a particular service. You can check the size of a packet's data payload by using the **dsize** option. This option takes a number as an argument. In addition, you can specify an upper bound by using the < operator, or you can choose a lower bound by using the > operator. Upper and lower bounds can be expressed with <>. For example:

alert tcp any any -> any any (msg:"Possible exploit"; content:"|90|"; \ offset:40; depth:75; dsize: >6000;)

This modifies the previous rule to match only if the data payload's size is greater than 6000 bytes, in addition to the other options criteria.

**The content-list Keyword**

The content-list keyword is used with a file name. The file name, which is used as an argument to this keyword, is a text file that contains a list of strings to be searched inside a packet. Each string is located on a separate line of the file. For example, a file named “Student” may contain the following two lines:

* + “Name”
  + “ID”

The following rule will search these strings in the data portion of all packets matching the rule criteria.

alert ip any any -> 192.168.1.0/24 any (content-list: \ "Student"; msg: "Student word matched";)

You can also use the negation sign! with the file name if you want to generate an alert for a packet where no strings match.

**Task 8:** Write a snort rule to generate an alert when opening SEECS homepage in a web browser. You can use keywords in the page content and use the content-list option. Write down the rule you used and copy-paste here the alert generated as a result in the alerts file. [1 Mark]

|  |
| --- |
| Alert Generation |
| alert tcp any any -> 10.2.31.67/24 any (content: "madrid"; msg: "hala madrid "; sid:1131;) |

To check the TCP flags of a packet, Snort provides the **flags** option. This option is especially useful for detecting port scans that employ various invalid flag combinations. For example, this rule will detect when the SYN and FIN flags are set at the same time:

alert any any -> any any (flags: SF,12; msg: "Possible SYN FIN scan";)

Valid flags are S for SYN, F for FIN, R for RST, P for PSH, A for ACK, and U for URG. In addition, Snort lets you check the values of the two reserved flag bits. You can specify these by using either 1 or 2. You can also match packets that have no flags set by using 0. There are also several operators that the **flags** option will accept. You can prepend either a + , \*, or ! to the flags, to match on all the flags plus any others, any of the flags, or only if none of the flags are set, respectively.

**Task 9**: Write a rule to generate an alert if the data size of an IP packet is larger than 6000 bytes. [1 Mark]

|  |
| --- |
| Alert Generation |
| alert ip any any -> 10.2.31.67/24 any (dsize: > 6000 ; msg: "hala madrid"; sid:234;) |

**Task 10**: Write a rule that generates an alert if a rule detects any scan attempt using SYN-FIN TCP packets. [1 Mark]

|  |
| --- |
| Alert Generation |
| alert tcp any any -> 10.2.31.67/24 any (msg:"hala madrid";flags:SF,12; sid:144; ) |

One of the best features of Snort is that it provides many plug-ins that can be used in the options field of a rule. The options discussed here should get you off to a good start. However, if you want to write more complex rules, consult Snort's excellent rule documentation, which contains full descriptions and examples for each of Snort's rule options. The Snort User's Manual is available at

<http://www.snort.org/docs/writing_rules/>.

**Deliverables**

Compile a single Word document by filling in the solution/answer part (as directed) along with the snapshots. Name your submission file as given below and submit this Word file on LMS before the deadline.

**Hadaiq Ahmad – 112807. – Bscs-4A Gp-1**

**Grade Criteria**

This lab is graded. Min Marks: 0. Max Marks: 10.

|  |  |  |
| --- | --- | --- |
| **Activity** | **Minimum** | **Maximum** |
| Documentation with a clearly defined understanding of the lab task and approach | Fail | Pass |
| Task 1 | 0 | 1 |
| Task 2 | 0 | 1 |
| Task 3 | 0 | 1 |
| Task 4 | 0 | 1 |
| Task 5 | 0 | 1 |
| Task 6 | 0 | 1 |
| Task 7 | 0 | 1 |
| Task 8 | 0 | 1 |
| Task 9 | 0 | 1 |
| Task 10 | 0 | 1 |