2010 International Conference on Networking and Digital Society

The Study on Network Intrusion Detection System of Snort

Zhou Zhimin, Chen Zhongwen , Zhou Tiecheng, Guan Xiaohui
Department of Computer Science Zhejiang Water Conservancy And Hydropoeer College
Hangzhou, China
{zhouzhm@zjwchc.com, chenzw@zjwchc.com, zhoutch@zjwchc.com, guanxh@zjwchc.com }

Abstract—**Network security is a complex and systematic project. The intrusion detection system is the first line of defense against network security. Snort is a famous intrusion detection system in the field of open source software. It is widely used in the intrusion prevention and detection domain in the world. In this paper, we explain how Snort implements the intrusion detection, which includes building the compiling environment and analysizing the work-flow and rule tree. This paper will provide a valuable reference for the study of Snort.**

Keywords-Network Security;Intrusion Detection;Snort

I. INTRODUCTION

With the fast development of computer technology, the Internet has permeated all aspects of everyday life. The followed network security has become a urgent problem to be addressed. Now various network security tools have been brought up, such as firewall, antivirus, etc. But there are still many security risks in the network[1]. The intrusion detection is the primary and important task in the network security system. It collects the sensitive information of network flow and warns for the possible attacks. The people can reinforce our network contrapuntally and purposefully basing on all kinds of forecast information. Thereby this will build a secure network environment. This paper proposes the Snort's implementation and application from the practical point of view. It includes building the compiling environment of Snort on Windows and analyzing the Snort's work flow and rule tree structure. We hope it can stimulate the discussion about Snort--a lightweight intrusion detection system with the colleagues who are interested in the network security.

II. THE SUMMARY OF SNORT

Snort is a lightweight network intrusion detection system, which is written with C language. It arised in 1998 and experienced a constant revise and perfection for more than a decade. it is open source. Now it has become a world-wide network intrusion detection and prevention software. Snort can strongly analyzes data flow and protocol in real time. It can be downloaded from the Internet and runs on almost all hardware platforms and operating systems. The flow work of Snort is composed of six parts: catching data package, analyzing code of data, preprocessing the package, parsing the rule, detecting the engine and logging [2]. Besides the Snort is simple, short and has good programming style. It is structured and easy to read. The latest version of Snort is 2.8.5.2. The following subject of Snort is based on the version 2.8.5.2.

III. THE IMPLEMENTATION OF SNORT

Before, the setup of Snort is very complex process, which involves detecting the integrity of compiling environment and the setup of Apache, Mysql, PHP, ADODB and ACID components[3]. Now, the Snort official website has provided some simple installation guides for Windows XP、Solaris 10 (SPARC) and Linux. We can quickly and easily build our intrusion detection system according to the guide of installation documents.

Taking into account the stability and security of operating system, we refer the Snort_Base_Minimal.pdf[4] to build Snort, Apache, SSL, PHP, MySQL, BASE and NTOP on Linux. The following is the installation steps:

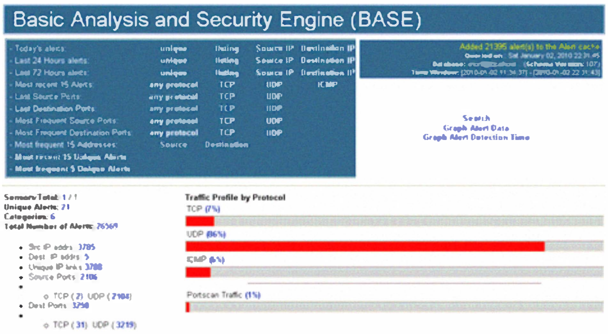
1) *Setup the operating system with minimal model.*
2) *Make the unnecessary services disable.*
3) *Setup the compiling environment using "yum" command and install the Apache, Mysql and PHP component.*
4) *Download and install the Snort source code and rule.*
5) *Modify the profile of Snort.*
6) *Build the Snort database with Mysql.*
7) *Install the ADODB and BASE.*
8) *Start all the required service.*

Snort monitors the network in the bypass mode. It catches the suspected data which attaches the Intranet. The monitoring result is shown on the Basic Analysis and Security Engine (BASE), which can intuitively analyze the catched data and display it. So the network security was improved by the reference data of Snort.

Figure 1 is the home of BASE. The BASE mainly shows the statistical result by time, IP address and port. We can find SSH login test, SQL detecting overflow, ICMP redirect host and other malicious acts as soon as the Snort is setup.

Figure 2 is the detail information of warning for a target IP.

978-1-4244-5161-6/10/$26.00 ©2010 IEEE



Basic Analysis and Security Engine (BASE)

- Today's alerts: unique listing Source IP Destination IP
- Last 24 Hours alerts: unique listing Source IP Destination IP
- Last 72 Hours alerts: unique listing Source IP Destination IP
- Most recent 15 Alerts: any protocol TCP UDP ICMP
- Last Source Ports: any protocol TCP UDP
- Last Destination Ports: any protocol TCP UDP
- Most Frequent Source Ports: any protocol TCP UDP
- Most Frequent Destination Ports: any protocol TCP UDP
- Most frequent 15 Addresses: Source Destination
- Most recent 15 Unique Alerts
- Most frequent 5 Unique Alerts

Added 21395 alert(s) to the Alert cache
Queried on: Sat January 02, 2010 22:31:45
Database: [redacted] (Schema Version: 107)
Time Window: [2010-01-02 11:34:37] - [2010-01-02 22:31:43]

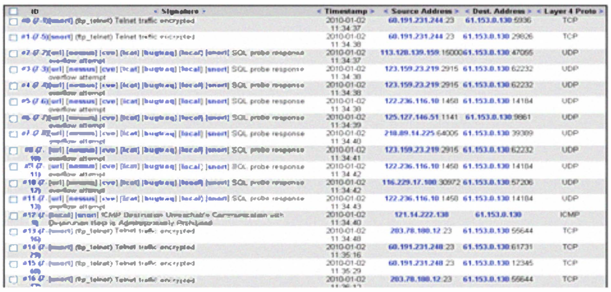
Search
Graph Alert Data
Graph Alert Detection Time

Sensors/Total: 1 / 1
Unique Alerts: 21
Categories: 6
Total Number of Alerts: 76569

- Src IP addrs: 3785
- Dest. IP addrs: 5
- Unique IP links 3788
- Source Ports: 2106
-
  - TCP (2) UDP (2104)
- Dest Ports: 3250
-
  - TCP (31) UDP (3219)

Traffic Profile by Protocol
TCP (7%)
UDP (86%)
ICMP (6%)
Portscan Traffic (1%)

Figure 1. The home of BASE



□ ID	Signature	Timestamp	Source Address	Dest. Address	Layer 4 Proto
□ #0-(7-1)	[snort] (ftp_telnet) Telnet traffic encrypted	2010-01-02 11:34:37	60.191.231.244:23	61.153.0.130:5936	TCP
□ #1-(7-5)	[snort] (ftp_telnet) Telnet traffic encrypted	2010-01-02 11:34:38	60.191.231.244:23	61.153.0.130:29826	TCP
□ #2-(7-7)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:37	113.128.139.159:15000	61.153.0.130:47066	UDP
□ #3-(7-3)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:38	123.159.23.219:2915	61.153.0.130:62232	UDP
□ #4-(7-4)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:38	123.159.23.219:2915	61.153.0.130:62232	UDP
□ #5-(7-6)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:38	122.236.116.10:1450	61.153.0.130:14184	UDP
□ #6-(7-7)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:39	125.127.146.51:1141	61.153.0.130:9861	UDP
□ #7-(7-8)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:40	218.89.14.225:64006	61.153.0.130:39309	UDP
□ #8-(7-10)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:41	123.159.23.219:2915	61.153.0.130:62232	UDP
□ #9-(7-11)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:42	122.236.116.10:1450	61.153.0.130:14184	UDP
□ #10-(7-12)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:42	116.229.17.100:30972	61.153.0.130:57206	UDP
□ #11-(7-13)	[url] [nessus] [cve] [icat] [bugtraq] [local] [snort] SQL probe response overflow attempt	2010-01-02 11:34:43	122.236.116.10:1450	61.153.0.130:14184	UDP
□ #12-(7-9)	[local] [snort] ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited	2010-01-02 11:34:40	121.14.222.138	61.153.0.130	ICMP
□ #13-(7-16)	[snort] (ftp_telnet) Telnet traffic encrypted	2010-01-02 11:34:48	203.78.100.12:23	61.153.0.130:55644	TCP
□ #14-(7-29)	[snort] (ftp_telnet) Telnet traffic encrypted	2010-01-02 11:35:16	60.191.231.248:23	61.153.0.130:61731	TCP
□ #15-(7-60)	[snort] (ftp_telnet) Telnet traffic encrypted	2010-01-02 11:35:29	60.191.231.248:23	61.153.0.130:12345	TCP
□ #16-(7-[illegible])	[snort] (ftp_telnet) Telnet traffic encrypted	2010-01-02	203.78.100.12:23	61.153.0.130:55644	TCP

Figure 2. The detail information of suspected attack

IV. BUILDING THE DEBUGGING ENVIRONMENT OF SNORT

You can analyze and modify the Snort in your familiar compiling environment, for example Microsoft VC++ integrated environment. The following is the steps of Snort in VC++:

A. *Installing Cygwin*

Cygwin is a simulation environment of Linux in Windows. It can transplant the application software from Linux to Windows. You don't have to install the Linux virtual machine. When you setup the Cygwin, please keep the default installation parameters. Or you will have to change the content of Snort project in VC++.

When you install the Cygwin, you should check the "Setup the bisoon related package" option, which is included in the Devel option. If you don't do this, there will be an error: "Can't find the command file" in the compiling process.

B. *Downloading the Snort source code*

After downloading and unzipping the Snort source code, you will compile it in VC++. Please modify the profile of Snort before running the Snort. This will make the Snort adapt the Windows. For more information, please refer the Snort official website.

V. ANALYZING SNORT CODE

A. *The working process of Snort*

The SnortMain implements the initialization and monitoring of Snort. Figure 3 is the working process.

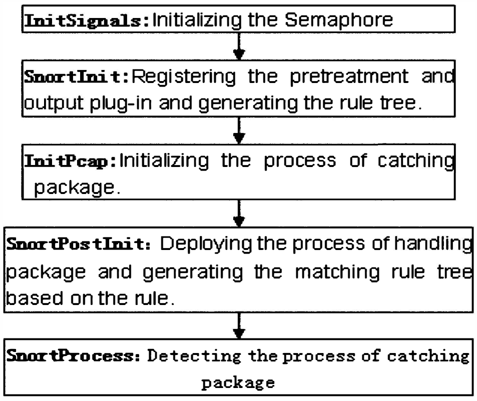

InitSignals: Initializing the Semaphore
SnortInit: Registering the pretreatment and output plug-in and generating the rule tree.
InitPcap: Initializing the process of catching package.
SnortPostInit: Deploying the process of handling package and generating the matching rule tree based on the rule.
SnortProcess: Detecting the process of catching package

Figure 3. The working process of Snort

B. *The rule tree of Snort*

The rule tree is the important data structure in Snort. It is helpful to grasp the rule matching of Snort. Figure 4 shows the structure of rule tree.

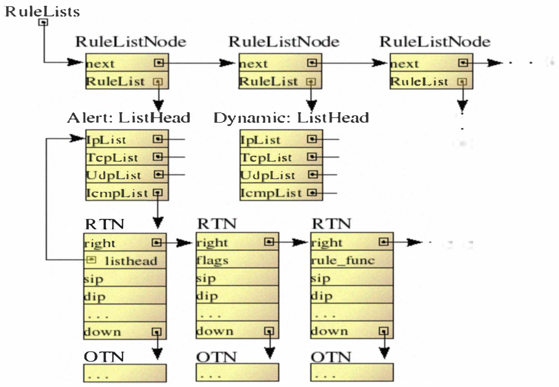


Figure 4. The structure of rule tree

From the figure 4, we can get that the node is classified into several lists according to the action. Each list is divided into four nodes according to the protocol type. Every node includes a RTN list basing on different IP and port. The different operations in the same RTN nodes form a OTN list. The whole lists construct the rule tree. Figure 5 shows the structure of rule tree. In this diagram, Snort firstly analyze the protocol type, IP address and port when catching a package before matching the rule. So the efficiency is very low. In version 2, Snort add a matching data structure to improve the matching efficiency. The revised structure of rule tree is shown in figure 6.

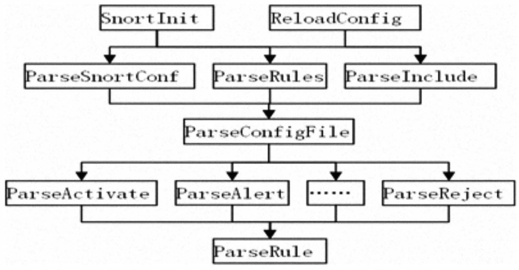


Figure 5. The function call of rule tree

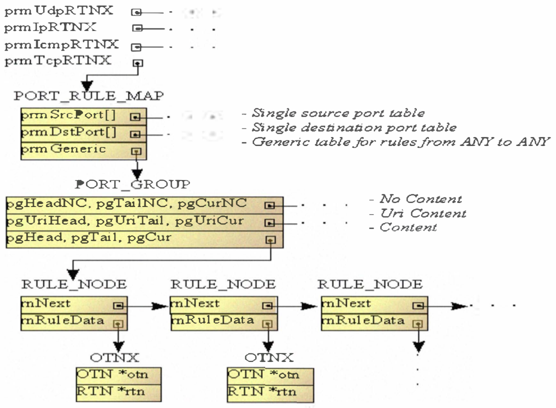


Figure 6. The fast matching structure of rule tree

The new rule tree includes four lists, and each list then is divided into several sub-lists according to the source port and destination port. Each sub-list is composed of three RULE_NODE lists, which combing the OTNX and early OTN list. The matching rule is in the way of the following figure 7.

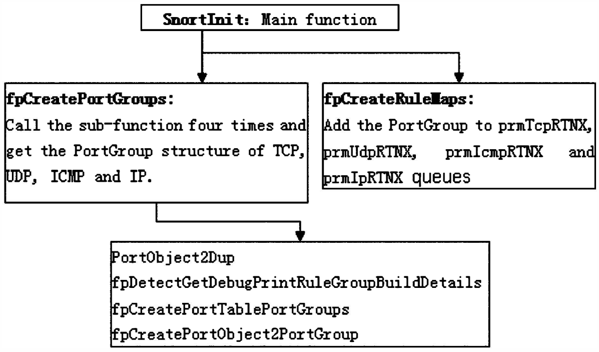

SnortInit: Main function
fpCreatePortGroups:
Call the sub-function four times and get the PortGroup structure of TCP, UDP, ICMP and IP.
fpCreateRuleMaps:
Add the PortGroup to prmTcpRTNX, prmUdpRTNX, prmIcmpRTNX and prmIpRTNX queues
PortObject2Dup
fpDetectGetDebugPrintRuleGroupBuildDetails
fpCreatePortTablePortGroups
fpCreatePortObject2PortGroup

Figure 7. The function call of fast rule tree

C. *Detecting the package in Snort*

The function call of detecting package is showed in figure 8. The function of SnortProcess calls IpqLoop, IpfwLoop or InterfaceThread function according to different parameters to grasp the package, pretreat, match the pattern and output data. The rule matching of data package is done by fpEvalHeaderSW function. Once the rule is matched correctly, Snort will output the warning.

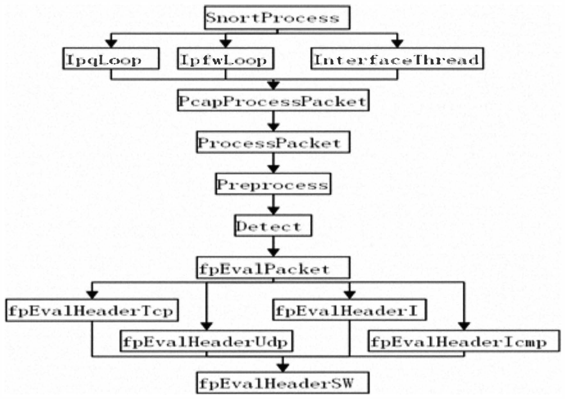


Figure 8. The function call of detecting package

VI. CONCLUSIONS

Snort is a common intrusion detection system. This paper proposes the process of Snort in Linux. It firstly shows how to debug the Snort in VC++ and then analyzes the important data structure and working process. The building process of intelligent matching rule and the working process of fpEvalHeaderSW also been explored.

ACKNOWLEDGMENT

This work is supported by Educational Commission of Zhejiang Province of China NO (20070849)

REFERENCES

[1] Yang Li. Research and Implementation of intrusion detection system based on Snort[J]. Beijing: The Technology and Application of Network Security .2009.11.

[2] JX Xu. The analysis of intrusion detection softwares. Journal of Southwest Guizhou Teachers College for Nationalities.[J].2008.3

[3] FF Wu. The Solution of Snort intrusion detection system[M]. China Machine Press. 2005. P86-110

[4] Patrick Harper. Snort and BASE Install on CentOS 4, RHEL 4 or Fedora Core [EB/OL] . http://assets.sourcefire.com/snort/setupguides/Snort_Base_Minimal.pdf

PROFILE

ZHOU Zhimin is an associate professor in department of computer science of Zhejiang Water Conservancy and Hydropoeer College. She focuses on the research of education.