Optimizing Rule on Open Source Firewall Using Content and PCRE Combination

Gandeva Bayu Satrya and Soo Young Shin

***Abstract*—In accordance with current technological developments, such as the Internet of Things, Big Data, and 5G Technology that almost all of them are better in terms of infrastructure, faster in terms of bandwidth, and more technological savvy than previous technology. From all of these developments, there are positive and negative impacts. The positive impact certainly gives things that can help our daily lives even to do business. While the negative impact is the most feared thing in almost every country such as theft of intellectual property, financial fraud, damage of service network, etc. And that are all we know of cybercrime.**

NIDS (Network Intrusion Detection System) is a software application that can see deeper into the network to prevent suspicious activity such as malicious software or unauthenticated activity. The contribution of this paper is to provide a NIDS system that is more optimal and inexpensive in implementation because it uses Open Source Firewall of cyber-attacks. With additional features such as content and PCRE (Perl Compatible Regular Expressions) that are already known, can provide the optimization to the network security system of a company / campus. Based on the implementation and testing that has done, recommendations of new safe system and policy are provided for small or large corporate in implementing NIDS.

***Index Terms*—Cybercrime, NIDS, signature-based, content, PCRE.**

I. Introduction

All of the latest internet technology like 5th Generation, Big Data, and Internet of Things are developed to satisfy users either in speed, bandwidth, and intelligence. The internet has created unparalleled opportunities for entertainment, education, research, trade, and even the military. With the presence of these new technology, the network security expert and the white hat community also always develop security features to combat the black hat community [1]-[3]. If viewed from a good point, the Internet can be used as the window of our lives in satisfying curiosity and developing new things. However, currently the use of the internet has many negative effects or some network security experts call it acts of cybercrime. Cybercrime is an act of infringement and abuse of computer systems connected to the internet, which resulted in direct or indirect losses [4].

Interpol is the world's intelligence agencies that are committed to be the coordinating body on the detection and prevention of digital crime through INTERPOL Global Complex for Innovation (IGCI) which was established in Singapore since 2010 [4]. In the website, Interpol wrote that Cybercrime is the latest developments of criminal action, such as exploiting the network speed and misuse of internet network to carry out criminal acts that know no boundaries, either physical or virtual. Interpol has divided cybercrime into three major categories [4]: first, the attack to the computer (software / hardware) for example malicious software (malware), robot networks (botnet), and network intrusion. Secondly, financial crime such as online fraud, penetration of online financial services and phishing. The third is abuse such as exploitation, crimes against children.

OWASP is a non-profit organization focused on improving software security that helps in making decisions about risks of software security [5]. In their site, there may be some kind of attacks that are currently popular. However, the case study in this paper will be examining four types of attacks such as Remote Access Tool (RAT), a web-based Trojans, SQL injection, and brute force password attack. These attacks will be tested in a topology environment that has been made on Section IV.

The academic researchers and security experts are and will always be developing how to use the internet safely and comfortably in terms of both hardware and software [6]. This paper will discuss the use of open source software firewall that is more optimal in detecting some cyber threats by adding digital content and PCRE (Perl Compatible Regular Expressions). It is often called as NIDS (Network Intrusion Detection System) Optimization. More precisely, NIDS is a software/hardware made for detecting multiple attacks immediately and applied to the network. In his book, Ciampa wrote there are 4 types for monitoring/examining network which are anomaly-based monitoring, behavior-based monitoring, heuristic monitoring, and signature-based monitoring. This paper will be optimizing signature-based method to audit the network traffic.

II. Related Works

As already mentioned in his book (Chapter 1 & 7) Rash mentioned that Linux Operating System provides an iptables feature, and those iptables provide the basic functionality of IPS and IDS with Signature-Based method [6]. Tools that is used is psad. Each release of psad contains signature set that is always updated in every psad package archives. These signatures can be developed for everyone who wants them and updated in http://www.cipherdyne.org/psad/.

Manuscript received May 29, 2015; revised November 30, 2015. This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the Global IT Talent support program (NIPA-2014-H0904-14-1005).

Gandeva Bayu Satrya and Soo Young Shin are with the Department of IT Convergence Engineering, School of Electronic Engineering, Kumoh National Institute of Technology, Gumi, South Korea (e-mail: gandevabs@kumoh.ac.kr, wdragon@kumoh.ac.kr).

Ciampa also mentions in his book entitled Network Security Fundamentals, that firewalls are divided into two categories based on their performance [2]: passive and active. Supposing that someone wants to make a home safe is to install a lot of locks and plated gates. Such things are a passive way. We cannot always rely on the passive ways all the time. How about when we are resting. Applying an alarm is the appropriate part. The point is to combine passive with active firewalls (also called as Intrusion Detection System), it is expected to ward off the attacks whenever it occurred. There are several ways to create an active firewall (IDS), one of which is signature-based [7]-[9].

The author's reason to use signature-based method because the development of the Internet, the development of Internet Security, as well as the scientific development of the attacker is directly proportional or always updated. Typical such as this is perfect for applying a signature, where each application definitely has a signature that is unique, and we can change or replace it according to our needs. By combining the signature-based features and open source firewall, will generate the most effective and inexpensive firewall optimization when applied on a campus or corporate.

III. Firewall and Signature-Based Overview

A. Firewall

Firewall, which are often called as packet filters (Open Source Based), is a hardware or software that is designed to prevent malicious packet to enter or exit the computer [6]. In the implementation, firewall is divided into two types: Host-Based and Network-Based. Host-based firewall is implemented on hosts connected to the network, such as anti-virus or a built-in firewall of an Operating System, while network-based firewall is implemented at the network level, which is close to routers or servers, e.g. Mikrotik Firewall. This paper is focused on network-based open source firewall which is Snort added with content features and PCRE [10]. The picture below is an example of the application of network-based firewall.

B. NIDS

NIDS is one of the types of IDS. Similar to those described regarding the firewall, IDS has two types of implementation which are in the host and in the network [7]-[9]. Before discussing it much deeper, the difference between a firewall and IDS will be explained using analogy. In our lives, for example in our home, a firewall is like a door with a key in it. While IDS is an alarm bells that will give alerts to the owner if there is anything that violates the rule that has been made. Of course, implementing both of them will be better (as shown in Fig. 1). Here is a picture of the implementation of firewalls and IDS [8].

C. Signature-Based

The creator of Snort, Marty Roesch, took tcpdump ability to take the package, and added the ability to analyze the packets against a set of signature attacks. This signature typically contains important bits and pieces of an attack that IDS should look for incoming network packages and classified as "bad traffic". The downside of signature detection is that it only knows to look for attacks based on the list made by the network administrator. Based on major component owned by Snort, a signature is at the preprocessor, whereas Rule Option (which will use the content and PCRE) are on the Detection part.

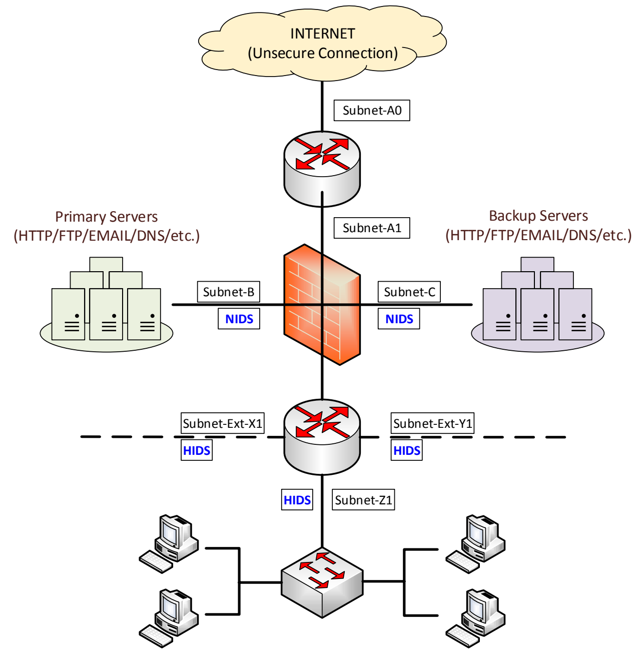


Fig. 1. Implementation firewall and NIDS.

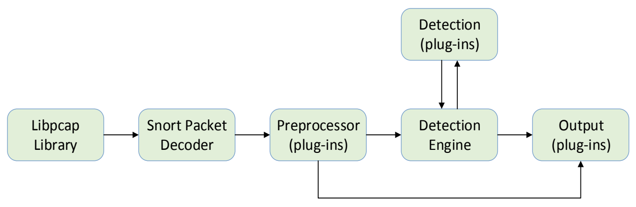


Fig. 2. Components of Snort [2], [3], [9].

D. Rule Option

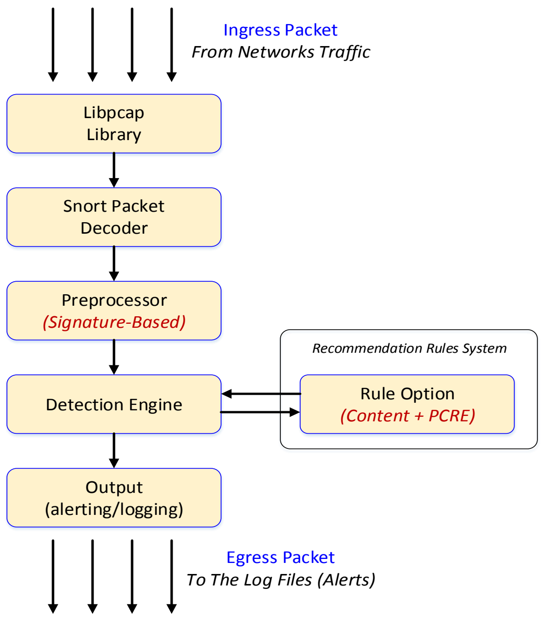
Snort has better features, because the program is not just an alarm for the attacker. In the designer did not want the simple "signature scanner". Snort create a "developer sets" for the network system security that can be modified to suit the wishes and the environment. Snort is a software based on "rule-based" engine. One of the best features of snort is Rule Option, where the rule can be made more optimal and right on target when a well-known target is needed. Based on Fig. 2 above, the position of Rule Option is in Detection Engine [7]-[9]. Then to detect data with dynamic nature, several choices of rule can be used such as rule content, IP options, TCP options, metadata options, and PCRE. This study focuses on the choice of content and PCRE [7], [11], [12].

IV. Recommendation System

A. Recommendation Rules System

Based on the explanations that were outlined in Section III, the following is the contribution made to snort Intrusion Detection System (as shown in Fig. 3). By using the Rule that contains Content and PCRE on Network Intrusion Detection System (NIDS) which built upon the comparison of bit /

strings that exist in the data packets attacks carried out repeatedly, it is expected to detect some kind of attack (attacker) on untrusted networks which are attacks that uses SQL Injection, RAT, and Trojan Downloader. For each scenario, more details can be seen on subsection C in this section.


Ingress Packet
From Networks Traffic
Libpcap Library
Snort Packet Decoder
Preprocessor (Signature-Based)
Detection Engine
Output (alerting/logging)
Recommendation Rules System
Rule Option (Content + PCRE)
Egress Packet
To The Log Files (Alerts)
Fig. 3. Recommendation rule option inside snort.

B. *Network Topology*

The modeling of network topology below is the design of signature-based analysis and implementation on Lan Network and DMZ Using content and PCRE. There are two clients on the trusted network, one client on untrusted network, and one attacker (who will test the rule system) used in this study as shown in Fig. 4. Firewall that is used is iptables on the Ubuntu Operating System. Here is the configuration of the network address that will be used.

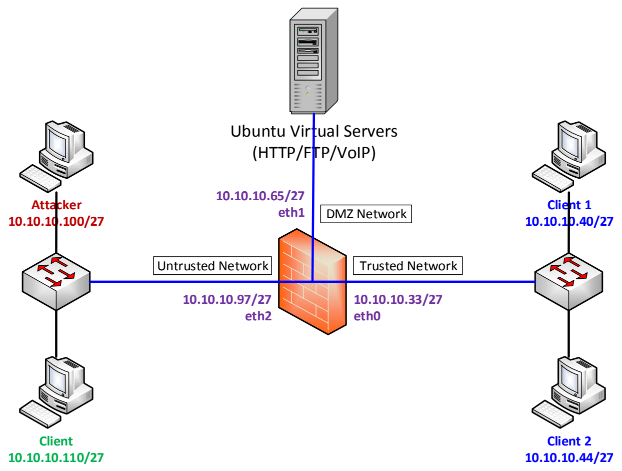


Fig. 4. Network topology test.

C. *Scenario System*

On this research, service test will be conducted on DMZ server and client on trusted network also the attacker on untrusted network, trough 3 attack scenarios with the types of attacks as follow:

1) *Attacks using Trojan horse or remote access tool (RAT)*

There are 5 types of Trojan horse that will be used in this research. Trojan are often used by attackers in virtual world for attacking and stealing the target's data. For examples Xtream RAT, and Cybergate RAT.

2) *Attacks using Trojan downloader (web-based Trojan)*

Trojan downloader is often used by attacker as the backdoor application for maintaining access to the attacked resources. This application also used for running other dangerous applications without the victim's knowledge. This research will analyze 2 types of Trojan downloader that often used by attackers.

3) *SQL injection attacks against websites with MySQL database*

SQL injection is a common attack to overwrite a website. This happens because of the lack of validation of input entered by the user. In this research, a SQL injection rule is modified and added so that the detection becomes more accurate.

V. Implementation and Evaluation

A. *Testing the Rules*

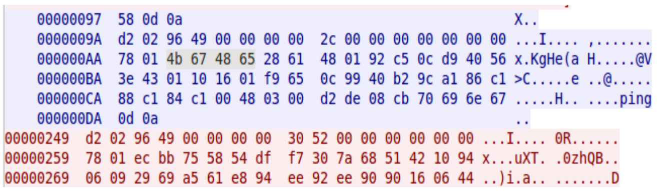
1) *Attack scenario using Trojan horse*

Trojan horse is application used to control a computer without the knowledge of the owner. After the computer was controlled, the attacker can freely explore the computer. For examples, to modify files, retrieve critical data, etc. Here are some analysis of Trojan horse.

a) *Xtream remote administration tools (RAT)*

Steps that are taken to analyze this Trojan until it resulted rules that will be implemented in IDP system are below:

1) A Do some attack simulation using Xtream RAT
2) Analyze the attack traffic using wireshark. Traffic analysis is aimed to look for the types of the Trojan, for example looking the data sent while the client is connected eith the Trojan server. The data that are fixed will be used as the standard. The analysis results are shown below in Fig. 5:



```
    00000097  58 0d 0a                                          X..
    0000009A  d2 02 96 49 00 00 00 00  2c 00 00 00 00 00 00 00 ...I.... ,.......
    000000AA  78 01 4b 67 48 65 28 61  48 01 92 c5 0c d9 40 56 x.KgHe(a H.....@V
    000000BA  3e 43 01 10 16 01 f9 65  0c 99 40 b2 9c a1 86 c1 >C.....e ..@.....
    000000CA  88 c1 84 c1 00 48 03 00  d2 de 08 cb 70 69 6e 67 .....H.. ....ping
    000000DA  0d 0a                                             ..
00000249  d2 02 96 49 00 00 00 00  30 52 00 00 00 00 00 00 ...I.... 0R......
00000259  78 01 ec bb 75 58 54 df  f7 30 7a 68 51 42 10 94 x...uXT. .0zhQB..
00000269  06 09 29 69 a5 61 e8 94  ee 92 ee 90 90 16 06 44 ..)i.a.. .......D
```

Fig. 5. Hexa file Xtreme RAT.

From data above, analysis can be determined by looking the data traffic between client and attacker, with requirements:

1) Traffic with red colour shown data traffic was sent from client to attacker.
2) Traffic with blue colour shown data traffic sent from attacker to client.

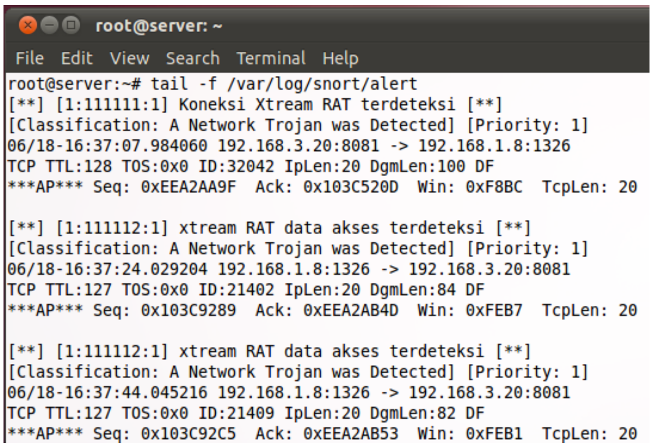
The analysis was began by looking at the data sent by attacker's computer to client's computer.

From both samples, can be seen there are static data which are data in the form of hexa code 4b 67 48 65 28 and string (a H. Both of the string always present when the simulation are done repeatedly. Therefore, these strings can be used as signature. Examples of rule resulted from those string (Fig. 6):

```
alert tcp any any -> any any (msg: "Xtream RAT connection is detected"; content:"|4b 67 48 65|(aH"; flow:established; pcre:"/\x4b\x67\x48\x65\x28/i"; classtype:Trojan-activity; sid:111111; rev:1;)
```

Fig. 6. Rule-1 Xtreme RAT connection.

After the rules are obtained, the last step was testing both of the rules on IDPS system. From figure below, can be seen that rules that have been made are capable to detect Trojan xtream RAT. The results are shown below Fig. 7:



```
root@server:~# tail -f /var/log/snort/alert
[**] [1:111111:1] Koneksi Xtream RAT terdeteksi [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/18-16:37:07.984060 192.168.3.20:8081 -> 192.168.1.8:1326
TCP TTL:128 TOS:0x0 ID:32042 IpLen:20 DgmLen:100 DF
***AP*** Seq: 0xEEA2AA9F  Ack: 0x103C520D  Win: 0xF8BC  TcpLen: 20

[**] [1:111112:1] xtream RAT data akses terdeteksi [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/18-16:37:24.029204 192.168.1.8:1326 -> 192.168.3.20:8081
TCP TTL:127 TOS:0x0 ID:21402 IpLen:20 DgmLen:84 DF
***AP*** Seq: 0x103C9289  Ack: 0xEEA2AB4D  Win: 0xFEB7  TcpLen: 20

[**] [1:111112:1] xtream RAT data akses terdeteksi [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/18-16:37:44.045216 192.168.1.8:1326 -> 192.168.3.20:8081
TCP TTL:127 TOS:0x0 ID:21409 IpLen:20 DgmLen:82 DF
***AP*** Seq: 0x103C92C5  Ack: 0xEEA2AB53  Win: 0xFEB1  TcpLen: 20
```

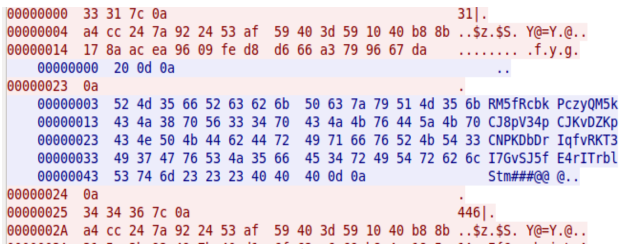
Fig. 7. Result 1: Rule vs Xtreme RAT.

b) *Cybergate remote administration tools (RAT)*

Steps of analysis that would generate rule are as follow:

1) Do some simulation using Trojan Cybergate.
2) Analyze attack traffic using wireshark.

Here are examples of rules generated during simulation, shown below in Fig. 8:



```
00000000  33 31 7c 0a                                       31|.
00000004  a4 cc 24 7a 92 24 53 af  59 40 3d 59 10 40 b8 8b ..$z.$S. Y@=Y.@..
00000014  17 8a ac ea 96 09 fe d8  d6 66 a3 79 96 67 da    ........ .f.y.g.
    00000000  20 0d 0a                                          ..
00000023  0a                                                .
    00000003  52 4d 35 66 52 63 62 6b  50 63 7a 79 51 4d 35 6b RM5fRcbk PczyQM5k
    00000013  43 4a 38 70 56 33 34 70  43 4a 4b 76 44 5a 4b 70 CJ8pV34p CJKvDZKp
    00000023  43 4e 50 4b 44 62 44 72  49 71 66 76 52 4b 54 33 CNPKDbDr IqfvRKT3
    00000033  49 37 47 76 53 4a 35 66  45 34 72 49 54 72 62 6c I7GvSJ5f E4rITrbl
    00000043  53 74 6d 23 23 23 40 40  40 0d 0a                Stm###@@ @..
00000024  0a                                                .
00000025  34 34 36 7c 0a                                    446|.
0000002A  a4 cc 24 7a 92 24 53 af  59 40 3d 59 10 40 b8 8b ..$z.$S. Y@=Y.@..
```

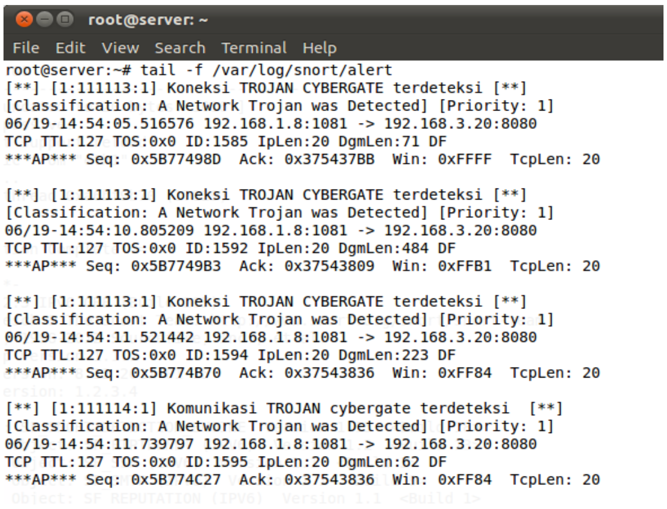
Fig. 8. Hexa file cybergate RAT.

Most of the strings above have different values. But there are some strings that are static including $ z, Y @ = Y. @. The data will always be there when the connection between the attacker and the client happened. String in the form of a period (.) Is not an exact representation of the actual code. The actual code can be seen on the left side of the above data, the characters in the form of Hexa code. Therefore, the writting of the rule does not use data in the form of ASCII strings, but using Hexa code. From those data, the hexa codes that are static and can be used as a signature is a4 cc 24 7a 92 24 53 af 59 40 59 10 40 b8 3d 8b (as shown in Fig. 9).
Rule generated from the signature is:

```
alert tcp any any -> any any (msg: " TROJAN CYBERGATE connection is detected"; content:"|a4 cc 24 7a 92 24 53 af 59 40 3d 59 10 40 b8 8b|"; flow:established; classtype:Trojan-activity; sid:111113; rev:1; )
```

Fig. 9. Rule-1 cybergate RAT connection.

3) The last step, testing rule that has been made on IDPS system, from the figure below can be seen that the rule can detect Trojan cybergate, the result as follow Fig. 10:



```
root@server:~# tail -f /var/log/snort/alert
[**] [1:111113:1] Koneksi TROJAN CYBERGATE terdeteksi [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/19-14:54:05.516576 192.168.1.8:1081 -> 192.168.3.20:8080
TCP TTL:127 TOS:0x0 ID:1585 IpLen:20 DgmLen:71 DF
***AP*** Seq: 0x5B77498D  Ack: 0x375437BB  Win: 0xFFFF  TcpLen: 20

[**] [1:111113:1] Koneksi TROJAN CYBERGATE terdeteksi [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/19-14:54:10.805209 192.168.1.8:1081 -> 192.168.3.20:8080
TCP TTL:127 TOS:0x0 ID:1592 IpLen:20 DgmLen:484 DF
***AP*** Seq: 0x5B7749B3  Ack: 0x37543809  Win: 0xFFB1  TcpLen: 20

[**] [1:111113:1] Koneksi TROJAN CYBERGATE terdeteksi [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/19-14:54:11.521442 192.168.1.8:1081 -> 192.168.3.20:8080
TCP TTL:127 TOS:0x0 ID:1594 IpLen:20 DgmLen:223 DF
***AP*** Seq: 0x5B774B70  Ack: 0x37543836  Win: 0xFF84  TcpLen: 20

[**] [1:111114:1] Komunikasi TROJAN cybergate terdeteksi  [**]
[Classification: A Network Trojan was Detected] [Priority: 1]
06/19-14:54:11.739797 192.168.1.8:1081 -> 192.168.3.20:8080
TCP TTL:127 TOS:0x0 ID:1595 IpLen:20 DgmLen:62 DF
***AP*** Seq: 0x5B774C27  Ack: 0x37543836  Win: 0xFF84  TcpLen: 20
```

Fig. 10. Result 2: Rule vs cybergate RAT.

2) *Attack scenario using Trojan downloader (web based Trojan)*

Trojan downloader is Trojan that uses website as intermediator. The infected computer will connect to a website that has been determined by the attacker. Computer that has been infected will be a zombie that controlled by attacker.

a) *Trojan Gbot loader*

Analysis of Trojan Gbot loader until IDPS system rule is generated, shown below:

1) Do attack simulation using Gbot Loader.
2) Take data traffic samples using wireshark.

Below (Fig. 11) are examples of data traffic result of Gbot Loader:

```
GET /bot/gbot/getcmd.php?id=-1068335068&traff=0 HTTP/1.1
User-Agent: Opera/9.80 (Windows NT 5.1; U; ru) Presto/2.6.30 Version/10.63
Host: 192.168.3.11
Connection: close
HTTP/1.1 200 OK
Date: Tue, 27 Jan 2015 05:44:25 GMT
Server: Apache/2.2.17 (Unix) mod_ssl/2.2.17 OpenSSL/0.9.8r DAV/2 PHP/5.3.6
X-Powered-By: PHP/5.3.6
Content-Length: 44
Connection: close
Content-Type: text/html
```

Fig. 11. Hexa file Trojan Gbot RAT.

Data above show connection after client's computer, which is infected with Trojan, connected to the server. Attacker cannot modify the traffic when Trojan has been connected. Attacker can only determine address of URL that will be connected first time by Trojan when it infecting computer. After that, the direction of communication are determined by Trojan. The parameter of getcmd.php is communication that is done by Trojan after it connected. It means getcmd.php always has fixed value.

3) Process of generating rule, are as follow:

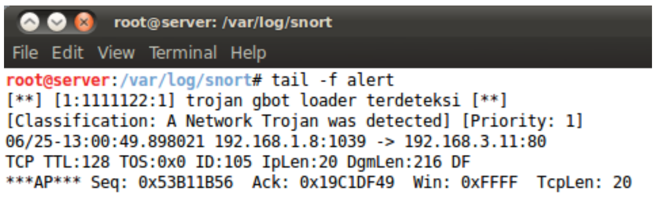
- Gbot uses user-agent opera 9.80(User-Agent: Opera/9.80) which is a component of http_header. Example of syntax rule is *content:"User|2d|Agent|3a| Opera/9.80"; http_header*
- Value of Id and traff always change.
- This attack classification on IDPS is Trojan-activity.

Example of resulted rule is shown in Fig. 12:

```
alert tcp any any -> any any (msg: "trojan gbot loader is
detected"; uricontent:"getcmd.php?id=";
content:"User|2d|Agent|3a| Opera/9.80"; http_header;
pcre:"/getcmd.php\?id\=\-\w+\x26traff\=.*/i";classtype:Troja
n-activity; sid:1111122; rev:1; )
```

Fig. 12. Rule Trojan Gbot RAT communication.

- After the rule has been generated, those rule are implemented on IDPS system to test the accuracy of the rule. From the result below (Fig. 13), can be seen rule that has been made can detect Trojan gbot loader. Here is the result of implementation of the rule:



```
root@server:/var/log/snort# tail -f alert
[**] [1:1111122:1] trojan gbot loader terdeteksi [**]
[Classification: A Network Trojan was detected] [Priority: 1]
06/25-13:00:49.898021 192.168.1.8:1039 -> 192.168.3.11:80
TCP TTL:128 TOS:0x0 ID:105 IpLen:20 DgmLen:216 DF
***AP*** Seq: 0x53B11B56  Ack: 0x19C1DF49  Win: 0xFFFF  TcpLen: 20
```

Fig. 13. Result 3: Rule vs Trojan Gbot loader RAT.

b) *Trojan elit loader*

Analyis of Trojan elit loader until rule of IDPS system is generated, shown below:

1) Do attack simulation using Elite Loader.
2) Sample data traffic using wireshark.

Here (Fig. 14) is example of result of data traffic from Elite Loader:

```
GET
/bot/elit/doit.php?v=3&id=c0528024-76487-640-239412
7-23295 HTTP/1.1
x-type: promake
User-Agent: Mozilla/4.0 (compatible; MSIE 6.1;
Windows XP)
Host: 192.168.3.11
HTTP/1.1 200 OK
Date: Tue, 27 Jan 2015 06:06:50 GMT
Server: Apache/2.2.17 (Unix) mod_ssl/2.2.17
OpenSSL/0.9.8r DAV/2 PHP/5.3.6
X-Powered-By: PHP/5.3.6
Content-Length: 1
Content-Type: text/html; charset=windows-1251
```

Fig. 14. Trojan Elite Loader in HTTP TCP stream.

The data above show the first connection established by Trojan to attacker's server. Parameter doit.php is uri parameter that is the first to b contacted by Trojan. Because it is the first connection, then the attacker can change the parameter. So, parameter doit.php is dynamic.

3) Process of generating rule, as follow:

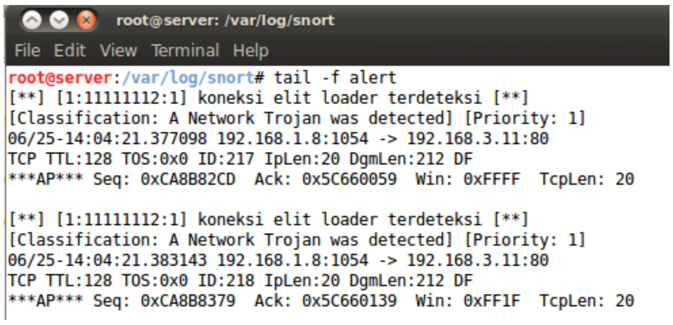
- Trojan use user-agent: Mozilla/4.0 (User-Agent: Mozilla/4.0 (compatible; MSIE 6.1; Windows XP)) and x-type: promake as http_header. Examples of IDPS syntax rule for detecting it are content:"User|2d|Agent|3a| Mozilla/4.0"; http_header; content:"x|2d|type|3a| promake"; http_header
- Value of v=3&id=c0528024-76487-640-2394127-23295 is parameter that can be detected with PCRE. For example, pcre:"/[a-z]+\.php\x3fv=\d+\x26id=.*/i"
- Classification of these attacks on IDPS is Trojan-activity.

Sample of generated rule is shown in Fig. 15:

```
alert tcp any any -> any any (msg: "Trojan Elit Loader
connection is detected"; content:"User|2d|Agent|3a|
Mozilla/4.0";http_header;content:"x|2d|type|3a|
promake";http_header;pcre:"/[a-z]+\.php\x3fv=\d+\x26id=.*/
i";classtype:Trojan-activity; sid:11111112; rev:1; )
```

Fig. 15. Rule-1 Trojan elite loader communication.

4) After the rule has been generated, the rule is implemented on IDPS system to test the accuracy of the rule. From the result below, can be seen the rule that has been made can detect Trojan elite loader. Here (Fig. 16) is the implementation result of the rule:



```
root@server:/var/log/snort# tail -f alert
[**] [1:11111112:1] koneksi elit loader terdeteksi [**]
[Classification: A Network Trojan was detected] [Priority: 1]
06/25-14:04:21.377098 192.168.1.8:1054 -> 192.168.3.11:80
TCP TTL:128 TOS:0x0 ID:217 IpLen:20 DgmLen:212 DF
***AP*** Seq: 0xCA8B82CD  Ack: 0x5C660059  Win: 0xFFFF  TcpLen: 20

[**] [1:11111112:1] koneksi elit loader terdeteksi [**]
[Classification: A Network Trojan was detected] [Priority: 1]
06/25-14:04:21.383143 192.168.1.8:1054 -> 192.168.3.11:80
TCP TTL:128 TOS:0x0 ID:218 IpLen:20 DgmLen:212 DF
***AP*** Seq: 0xCA8B8379  Ack: 0x5C660139  Win: 0xFF1F  TcpLen: 20
```

Fig. 16. Result 4: Rule vs Trojan elite loader.

3) *Skenario Serangan SQL Injection*

SQL injection is an attack that occurred on a website where the attacker input the SQL queries to the website so that those queries can be executed by database and show the result to attacker. SQL injection has the potential to read all content of datatbase, read the server configuration, etc.

Analysis of SQL injection until generating IDPS system rule are as follow:

1) Do simulation of SQL injection attack.
2) Take samples of data traffic using wireshark.

Here (Fig. 17) are samples of data traffic from dari SQL injection:

The data show that the attacker is trying to check whether the targeted website has SQL injection gap or no. After the gap has been found, attacker tried to input SQL syntax on the website to look for username and password from the database as shown by second data above. From the data above, can be obtained some static data including select, from. The sequence of those words shows the SQL injection signature.

3) Processes of generating rule are:

- All signature are inputted to PCRE format because the characters that exist before and after the words are dynamic.
- Check the uppercase and lowercase charecters that are written on hexa code for each signature. These checking can be written as:
- The word Select is written as (s|%73|%53)(e|%65|%45)(l|%6C|%4C)(e|%65|%45)(c|%63|%43)(t|%74|%45)
- The word From is written as (f|%66|%46)(r|%72|%52)(o|%6F|%4F)(m|%6D|%4D)

At the end of SQL injection attack, there is a closing character (--). These character also inputted into PCRE format. The resulted rule is shown in Fig. 18:

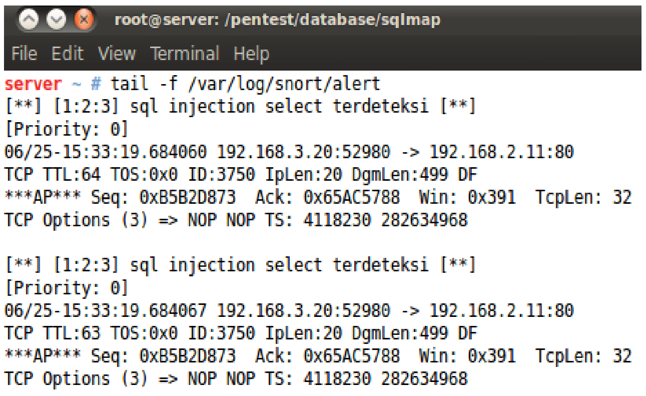
```
GET
/sqli/events.php?id=-7104%20UNION%20ALL%20SELECT%20
CONCAT%280x3a6676773a%2C0x4a4a47484e746f697565%2C
0x3a6364683a%29%2C%20NULL%20UNION%20ALL%20SEL
ECT%20CONCAT%280x3a6676773a%2C0x526f7a57524c6a566
96b%2C0x3a6364683a%29%2C%20NULL%23 HTTP/1.1
Accept-Encoding: identity
Accept-Language: en-us,en;q=0.5
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)
Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7
Host: 192.168.2.11
Pragma: no-cache
Cache-Control: no-cache,no-store
HTTP/1.1 200 OK
Date: Tue, 27 Jan 2015 08:30:24 GMT
Server: Apache/2.2.17 (Unix) mod_ssl/2.2.17 OpenSSL/0.9.8r
DAV/2 PHP/5.3.6
X-Powered-By: PHP/5.3.6
Connection: close
Content-Type: text/html
```

Fig. 17. SQL injection in HTTP TCP stream.

```
alert tcp any any -> any any  (msg: "sql injection select is
detected"; flow: established;
pcre:"/(s|%73|%53)(e|%65|%45)(l|%6C|%4C)(e|%65|%45)(c|%
63|%43)(t|%74|%45).*(f|%66|%46)(r|%72|%52)(o|%6F|%4F)(
m|%6D|%4D).*(\-\-|\/\*|\#)/i"; sid: 2; rev: 1;)
```

Fig. 18. Rule SQL injection communication.

4) After the rule is resulted, it is implemented on IDPS system to test its accuracy. From the result below, can be seen the rule that has been made can detect SQL injection attack. Here (Fig. 19) is example of the rule implementation:


root@server: /pentest/database/sqlmap
File Edit View Terminal Help

```
server ~ # tail -f /var/log/snort/alert
[**] [1:2:3] sql injection select terdeteksi [**]
[Priority: 0]
06/25-15:33:19.684060 192.168.3.20:52980 -> 192.168.2.11:80
TCP TTL:64 TOS:0x0 ID:3750 IpLen:20 DgmLen:499 DF
***AP*** Seq: 0xB5B2D873  Ack: 0x65AC5788  Win: 0x391  TcpLen: 32
TCP Options (3) => NOP NOP TS: 4118230 282634968

[**] [1:2:3] sql injection select terdeteksi [**]
[Priority: 0]
06/25-15:33:19.684067 192.168.3.20:52980 -> 192.168.2.11:80
TCP TTL:63 TOS:0x0 ID:3750 IpLen:20 DgmLen:499 DF
***AP*** Seq: 0xB5B2D873  Ack: 0x65AC5788  Win: 0x391  TcpLen: 32
TCP Options (3) => NOP NOP TS: 4118230 282634968
```

Fig. 19. Result 5: Rule vs SQL injection.

B. *Analyisis of the Rules*

TABLE I: COMMON FIREWALL VS SIGNATURE-BASED

Type of Attack	Hypotheses	Normal Iptables (IP & Port)	Signature-Based (Content & PCRE)
Xtream RAT	Success	Failed	Success (Detected)
Cybergate RAT	Success	Failed	Success (Detected)
Trojan Gbot loader	Success	Failed	Success (Detected)
Trojan Elit Loader	Success	Failed	Success (Detected)
SQL Injection	Success	Failed	Success (Detected)

According to 5 testing that have been conducted, if it were counted based on sensitivity and specificity, it should be 100% (shown in Table I). And it will be the opposite with conventional firewall that only based on IP and Port Number. According besic theory of Signature-Based, it is very suitable for obvious and known environment. And the result will definitely has high detection accuracy (probability of a positive test among packets with malware).

VI. CONCLUSION

Based on test results of the scenarios that has been created, can be concluded that all rules has been made are capable of detecting Trojan Horse/RAT, Trojan Downloader attacks, and SQL injection. And the uses of rule content and PCRE (Perl Compatible Regular Expression) on NIDS rules are very effective in detecting types of attacks that have been mentioned. There are many other attacks that have not been tried based on OWASP such as Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), Insecure Direct Object References, and etc. And can also be tried to be implemented in large network scale (with complex router) in order to obtain a more optimal detection time dan complexity situation.

ACKNOWLEDGMENT

This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the Global IT Talent support program (IITP-2015-R0618-15-1006) supervised by the IITP (Institute for Information and Communication Technology Promotion). Thank you for anonymous person who did not want to be mentioned in this paper, thank you for your cooperation and suggestion.

REFERENCES

[1] A. Philipp, D. Cowen, and C. Davis, *Hacking Exposed Computer Forensics Second Edition Reviews*, USA: The McGraw-Hill Companies, 2010.

[2] M. Ciampa, *CompTIA Security+ Guide to Network Security Fundamentals Paperback*, 5th ed. USA: Cengage Learning, 2014.

[3] M. T. Britz, *Computer Forensics and Cyber Crime: An Introduction*, 3rd ed., 2013.

[4] Interpol. Cybercrime. [Online]. Available: http://www.interpol.int/Crime-areas/Cybercrime/Cybercrime

[5] OWASP. 2013 Top 10 List. [Online]. Available: https://www.owasp.org/index.php/Top_10_2013-Top_10

[6] M. Rash, *Linux Firewalls: Attack Detection and Response with iptables, psad, and fwsnort Paperback*, 1st ed. UAS: No Starch Press, September 15, 2007.

[7] C. Scott, P. Wolfe, and B. Hayes, *Snort For Dummies Paperback*, July 9, 2004, For Dummies Wiley Publishing, Inc., Indianapolis, Indiana.
[8] J. Beale and B. Caswell, *Snort 2.1 Intrusion Detection*, 2nd ed. USA: Syngress Publishing, Inc., May 2004.
[9] R. U. Rehman, *Intrusion Detection with SNORT: Advanced IDS Techniques Using SNORT, Apache, MySQL, PHP, and ACID Paperback*, UAS: Pearson Education, Inc., May 18, 2003.
[10] PCRE. Perl regular expressions man page. [Online]. Available: http://www.pcre.org/
[11] J. E. F. Friedl, *Mastering Regular Expressions Paperback*, 3rd ed. USA: O'Reilly Media, August 15, 2006.
[12] M. Fitzgerald, *Introducing Regular Expressions Paperback*, 1st ed. USA: O'Reilly Media, August 3, 2012.

Gandeva B. Satrya is currently a Ph.D candidate of security communication in next generation networks, IT Convergence Engineering, School of Electronic Engineering, Kumoh National Institute of Technology, South Korea. He has been a lecturer and researcher at the Faculty of Informatics in Telkom University, Bandung, Indonesia since 2011. His areas of expertise are routing protocol, packet scheduling, and security communication in next generation network. He previously worked as a BSS engineer at PT. ZTE Indonesia from June 2008 to December 2010. He completed his master's degree in informatics engineering majoring media informatics at IT Telkom, Bandung, Indonesia in June 2012 and received his bachelor's degree also in informatics engineering from STT Telkom in March 2008.

Soo Y. Shin received his B.S., M.S., and Ph.D degrees in electrical engineering and computer science from Seoul National University, Korea in 1999, 2001, and 2006, respectively. His research interests include wireless LAN, WPAN, WBAN, wireless mesh network, sensor networks, coexistence among wireless networks, industrial and military network, cognitive radio networks, MIMO, OFDM, mmWave, NOMA and next generation mobile wireless broadband networks (4G/5G). He was a visiting scholar in FUNLab at University of Washington, US, from July 2006 to June 2007. After 3 years working in WiMAX Design Lab. of Samsung Electronics, he is now an assistant professor in School of Electronics in Kumoh National Institute of Technology since September 2010.