Wireless Pers Commun
DOI 10.1007/s11277-016-3209-9

Performance Comparison and Detection Analysis in Snort and Suricata Environment

Wonhyung Park[1] · **Seongjin Ahn**[2]

Abstract Recently, crimes are cause in the internet by hacking to target one's and the companies financial. Due to the massive crimes that are caused by digital convergence and ubiquitous IT system, it is clear that the amount of network packet which need to be processed are rising. The digital convergence and ubiquitous IT system caused the IDS (Intrusion Detection System) to process packets more than the past. Snort (version 2.x) is a leading open source IDS which has a long history but since it was built a long time ago, it has several limitations which are not fit for today's requirements. Such as, it's processing unit is in single threading. On the other hand, Suricara was built to cover Snorts these disadvantages. To cover massive amount of packets which are caused by digital convergence and ubiquitous IT system Suricata's have the availability to process packets in multi-threading environment. In this paper we have analyzed and compared Snort and Suricata's processing and detection rate to decide which is better in single threading or multi-threading environment.

Keywords Snort · Suricata · Network security · Detection system

1 Introduction

Since the network threat and bandwidth are rising the trend of IDS are moving from single threading techniques to multi-threading techniques to gain better performance in multi core hardware's. Among those open source IDS, Snort is known to be the best single threading

✉ Seongjin Ahn
sjahn84@gmail.com

Wonhyung Park
whpark@kdu.ac.kr

[1] Department of Cyber Security, Far East University, Wangjang-ri, Gamgok-myeon, Eumseong-gun, Chungcheongbuk-do, Republic of Korea

[2] Department of Computer Education, Sungkyunkwan University Seoul, Seoul, Republic of Korea

IDS. The reason why Snort has the biggest market share is because it has great stability and is good at detecting malicious packets. However, in the future, bigger bandwidth of network packets will cause single threaded IDS to have limitation on processing them. Thus, the users will slowly begin to move to multi-threaded IDS. Suricata is the open source multi-threaded IDS system which is compatible with Snort rules. Since Snort has the best market share but have limitations to its processing due to single thread, it is necessity to analyze and compare detection and performance of Snort and Suricata. To give better user experience to choose from single thread IDS to multi thread.

2 Related Work

Snort and Suricata are both open source and are capable of running NIDS (Network Based Intrusion Detection System) and HIDS (Host Based Intrusion Detection System). In this chapter we will talk about how and what features Snort and Suricata have.

2.1 Snort

Snort has been developed to perform its role as a packet sniffer (Packet Sniffer) by Roesch [1] in November 1998. Snort's has four architectures which contains Packet Sniffer, Pre-Processor, Detection Engine, Alerting or Logging [2].

First, Packet Sniffer eavesdrops on network traffic (network backbone). Second Pre-Processor checks the packets to determine if the packets are in certain behavior [3, 4]. Third, the Detection Engine takes incoming packets and matches them through rules. Fourth, when the rules match, the Alert or Logging takes place whether log to the database, or alerts the user [5, 6, 7] (Fig. 1).

2.2 Suricata

Suricata has been developed in 2010 by the OISF (Open Information Security Foundation) which received financial support from the US Department of Homeland Security [8]. Suricata's architecture are extremely similar to snort, but rather than using single thread, Suricata implements multi thread to process packets [9]. This allows unique feature of Suricata, which maximizes the ability to receive packet. Since Snort was single

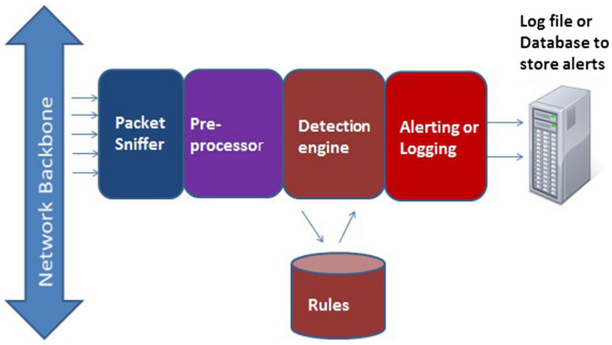


Fig. 1 Snort architecture

threaded, when the packets overflew the ability to receive bandwidth, Snort ignored the packets received. Thus Suricata has one of the beneficial feature by multi-threading. As seen on the picture below, Suricata have multiple threads of detection engine [10, 11] (Fig. 2, Table 1).

2.3 Security Onion

Security Onion is an OS (Operating System) which are capable of easily installing Snort or Suricata by users choice [12]. It was developed by Doug Burks. The biggest benefit of Security Onion is its easy use of installing Snort and Suricata. Also it can install Sguil, Squert to monitor and can install the OS as a sensor or the server itself [13]. The recommended RAM is 16 GB and minimum requirement of RAM is 3 GB [14]. In this test we used the version 12.04.04 and used 64-bit OS, and RAM of 3 GB to test the IDS/IPS in extreme environment. Also we have installed the Security Onion as a standalone server, installed Sguil, Snortby, Elsa and set the rules as Emerging Threats GPL ruleset (Fig. 3).

3 Detection Analysis in Snort and Suricata

Suricata is built for replacing Snort's disadvantage which have low bandwidth of packet in multi core environment. This information must be research to find out which product performs better in single or multi core environment. To find out which product performs better we have analysis Snort and Suricata's detection and performance rate. Suricata has the latest technology and was built to replace Snort. Thus Suricata has similar rule syntax with Snort and can be top-compatible. Also it is capable of GPU acceleration, HTTP parsing, and more. In this content we have analyzed and compared Snort and Suricata detection and performance to find which product is better.

3.1 Prepare Method in IDS and IPS

Verification test system configuration can be composed of two.

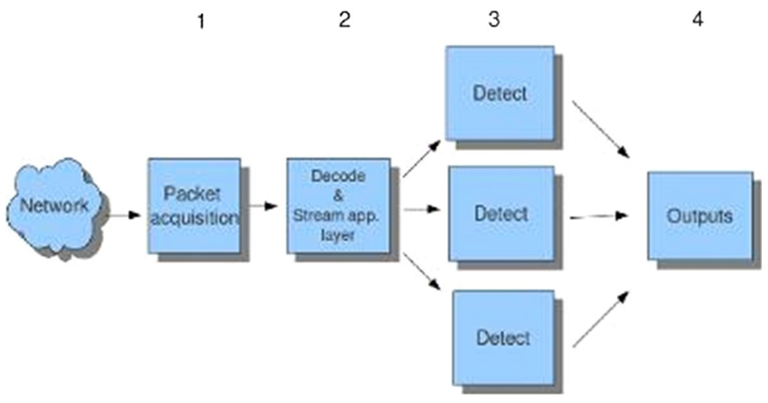


Fig. 2 Suricata architecture

Table 1 Snort/sruricata feature

Function	Snort	Suricata
IPv6	Added later	O
IP reputation	X	O
Automated protocol detection	X	O
GPU acceleration	X	O
Advanced HTTP parsing	X	O
HTTP access logging	X	O
SMB access logging	X	O
HTTP block list lookups	X	O

Fig. 3 Windows of security onion

First of all Frame-Work and IDS/IPS (Intrusion Detection System/Intrusion Prevention System) are on 1:1 structure. This system is called HIDS (Host Intrusion Detection System). HIDS layout is installing IDS/IPS in the victim server. These causes the packets not to cross firewall, networks, services, middle part which is possible to obtain a subtle direct result.

Second, Service (Web-Server, FTP, SSH, etc.) and Network IDS/IPS (Network Intrusion Detection System/Network Intrusion Prevention System), Framework's 1:1:1 structure. This structure is differenced by HIDS by the server and processing by the response to the client's request independently of the form of the IDS. Also this structure is to conduct the test in a similar state to the actual operating environment. However, we need at least one more node then the HIDS structure. Also it has the danger of network instability which cause it difficult to obtain a subtle direct result. Therefore, few errors should be considered to perform the right test (Fig. 4, Table 2).

In this test we have tested detection rate and performance rate using HIDS structure to get subtle result. Also to get accurate results we have disabled all the firewalls and installed the victim and the attacker on the same network.

3.2 Results of Detection Analysis

3.2.1 Results in Single Core

Snort's detection rate in single core have detected a few attacks. These results are because the single core environment makes it difficult to process the packets. The picture below is the Snorts content-specific attack detection (Fig. 5).

Suricata, in single core environment, lots of attacks were detected. These results are conflicting results with Snort. The picture below is the Suricata's content-specific attack detection (Fig. 6).

The figure below shows the overall results of the detection Snort and Suricata on a single core (Fig. 7).

3.2.2 Results in Multi Core

Snort in multi core environment have detected only a few attacks. These results can show that the Snort only supports single threading and could not use the benefit of the multi core environment. Low-bandwidth packet throughput cause some logs cannot be output correctly. The picture below is the Snorts content-specific attack detection (Fig. 8).

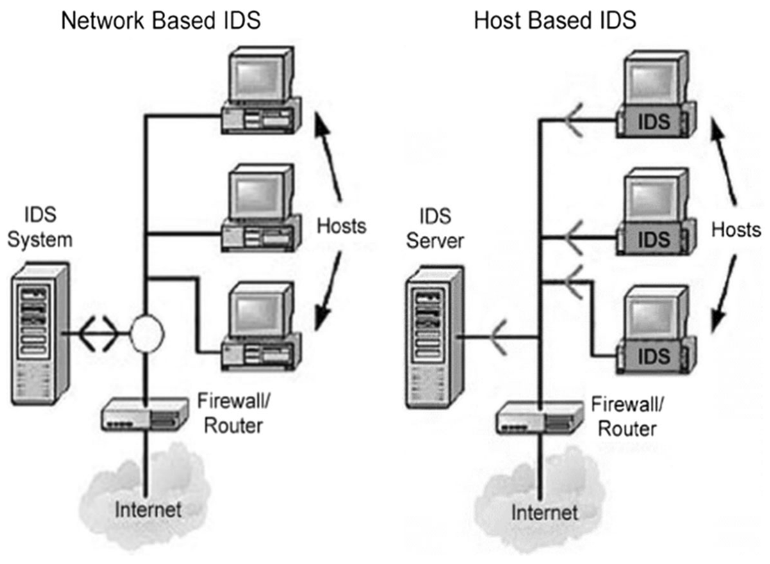
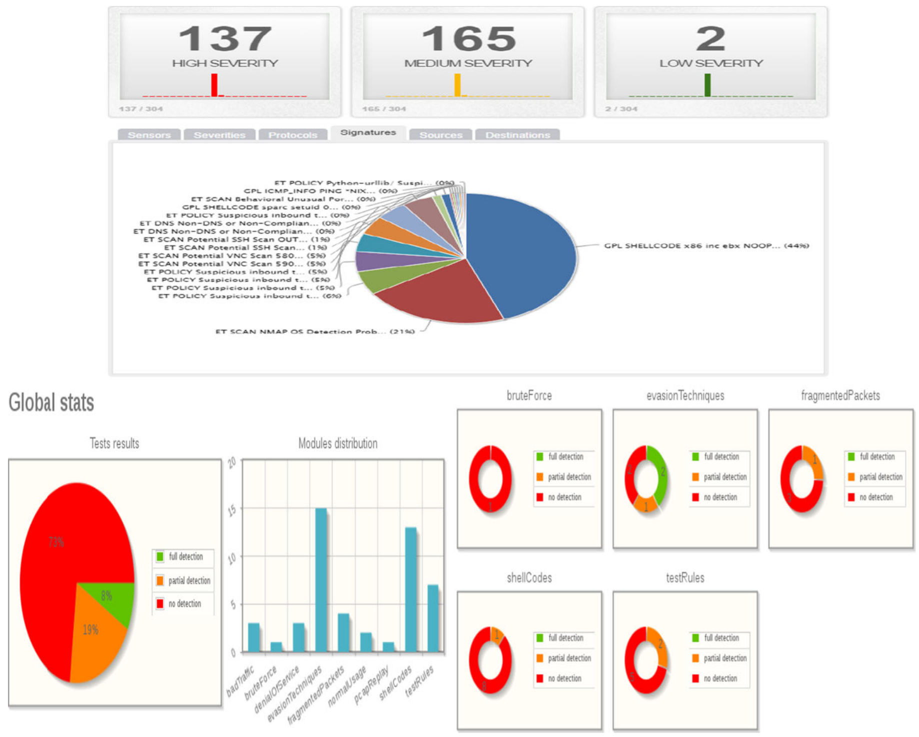


Fig. 4 NIDS and HIDS implementation

Table 2 Pytbull attack module

Attack module	Description
badTraffic	Non RFC compliant packets are sent to the server to test how packets are processed
bruteForce	Tests the ability of the server to track brute force attacks (e.g. FTP). Makes use of custom rules on Snort and Suricata
denialOfService	Denial of Service module. Will attempt to flood the server with special packets
evasionTechniques	Evasion Techniques module. Will send obfuscated attacks to the remote target
fragmentedPackets	Fragmented Packets module. Will fragment attacks on multiple small packets
pcapReplay	Test whether you can reconfigure the pcap file
shellCodes	ShellCodes module. Will send malicious hex-encoded payloads to the remote target
testRules	Test Rules module. Will test alerts based on signature files



137
HIGH SEVERITY
137 / 304
165
MEDIUM SEVERITY
165 / 304
2
LOW SEVERITY
2 / 304
Sensors
Severities
Protocols
Signatures
Sources
Destinations
ET POLICY Python-urllib/ Suspi... (0%)
GPL ICMP_INFO PING *NIX... (0%)
ET SCAN Behavioral Unusual Por... (0%)
GPL SHELLCODE sparc setuid 0... (0%)
ET POLICY Suspicious inbound t... (0%)
ET DNS Non-DNS or Non-Complian... (0%)
ET DNS Non-DNS or Non-Complian... (0%)
ET SCAN Potential SSH Scan OUT... (1%)
ET SCAN Potential SSH Scan... (1%)
ET SCAN Potential VNC Scan 580... (5%)
ET SCAN Potential VNC Scan 590... (5%)
ET POLICY Suspicious inbound t... (5%)
ET POLICY Suspicious inbound t... (5%)
ET POLICY Suspicious inbound t... (5%)
ET POLICY Suspicious inbound t... (6%)
GPL SHELLCODE x86 inc ebx NOOP... (44%)
ET SCAN NMAP OS Detection Prob... (21%)
Global stats
Tests results
73%
8%
19%
full detection
partial detection
no detection
Modules distribution
20
15
10
5
0
badTraffic
bruteForce
denialOfService
evasionTechniques
fragmentedPackets
normalUsage
pcapReplay
shellCodes
testRules
bruteForce
evasionTechniques
fragmentedPackets
shellCodes
testRules

Fig. 5 Snort single thread overview

Suricata in multi core environment has detected almost all attack packets. These results are conflicting results with Snort. The snorts availability to use multi core and multi-threading causes detection rate a bit better than the single core detection rate. The picture below is the Suricata's content-specific attack detection (Fig. 9).

The picture below is the Snort and Suricata's detection rate in multi core environment (Fig. 10).

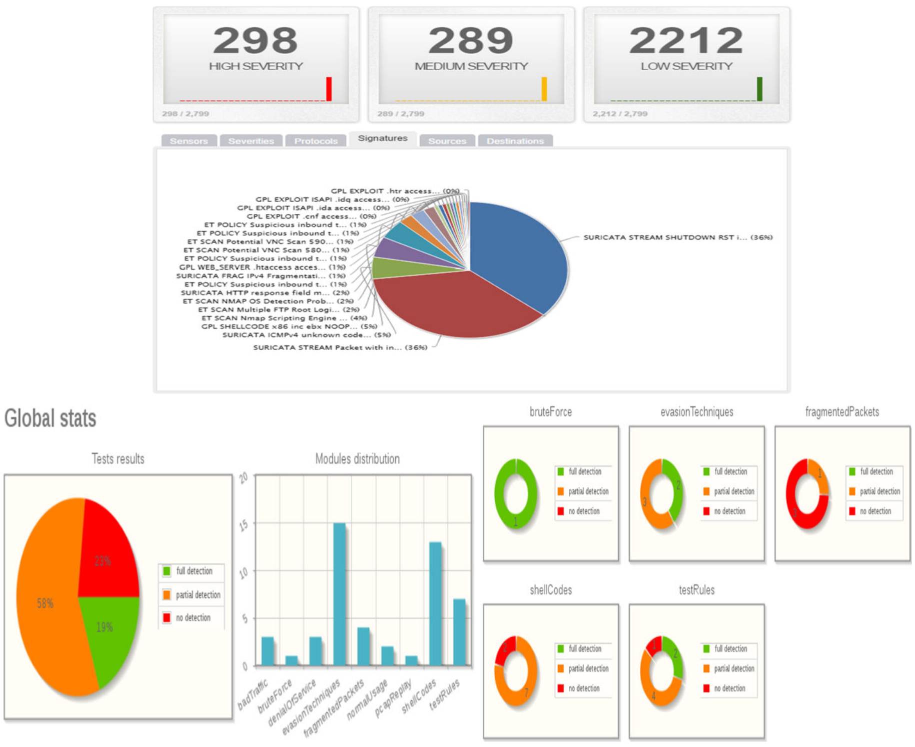
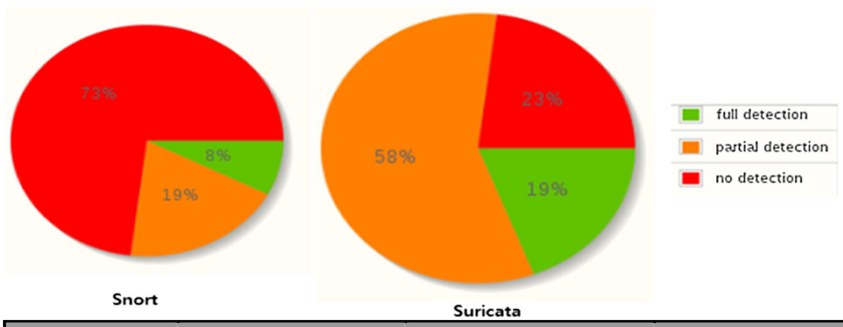

298
HIGH SEVERITY
298 / 2,799
289
MEDIUM SEVERITY
289 / 2,799
2212
LOW SEVERITY
2,212 / 2,799
Sensors
Severities
Protocols
Signatures
Sources
Destinations
GPL EXPLOIT .htr access... (0%)
GPL EXPLOIT ISAPI .idq access... (0%)
GPL EXPLOIT ISAPI .ida access... (0%)
GPL EXPLOIT .cnf access... (0%)
ET POLICY Suspicious inbound t... (1%)
ET POLICY Suspicious inbound t... (1%)
ET SCAN Potential VNC Scan 590... (1%)
ET SCAN Potential VNC Scan 580... (1%)
ET POLICY Suspicious inbound t... (1%)
GPL WEB_SERVER .htaccess acces... (1%)
SURICATA FRAG IPv4 Fragmentati... (1%)
ET POLICY Suspicious inbound t... (1%)
SURICATA HTTP response field m... (2%)
ET SCAN NMAP OS Detection Prob... (2%)
ET SCAN Multiple FTP Root Logi... (2%)
ET SCAN Nmap Scripting Engine ... (4%)
GPL SHELLCODE x86 inc ebx NOOP... (5%)
SURICATA ICMPv4 unknown code... (5%)
SURICATA STREAM Packet with in... (36%)
SURICATA STREAM SHUTDOWN RST i... (36%)
Global stats
Tests results
58%
23%
19%
full detection
partial detection
no detection
Modules distribution
0
5
10
15
20
badTraffic
bruteForce
denialOfService
evasionTechniques
fragmentedPackets
normalUsage
pcapReplay
shellCodes
testRules
bruteForce
evasionTechniques
fragmentedPackets
shellCodes
testRules

Fig. 6 Surcata single thread overview



IDS	High	Medium	Low
Snort	137	165	2
Suricata	298	289	2212

Fig. 7 Snort and suricata detection rate

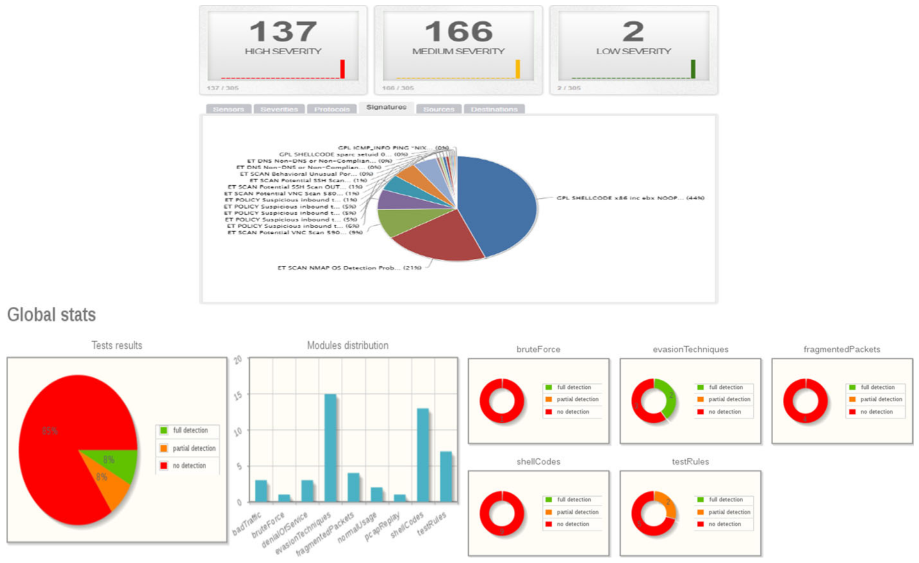

137
HIGH SEVERITY
137 / 305
166
MEDIUM SEVERITY
166 / 305
2
LOW SEVERITY
2 / 305
Sensors
Severities
Protocols
Signatures
Sources
Destinations
GPL ICMP_INFO PING *NIX... (0%)
GPL SHELLCODE sparc setuid 0... (0%)
ET DNS Non-DNS or Non-Complian... (0%)
ET DNS Non-DNS or Non-Complian... (0%)
ET SCAN Behavioral Unusual Por... (0%)
ET SCAN Potential SSH Scan... (1%)
ET SCAN Potential SSH Scan OUT... (1%)
ET SCAN Potential VNC Scan 580... (1%)
ET POLICY Suspicious inbound t... (1%)
ET POLICY Suspicious inbound t... (5%)
ET POLICY Suspicious inbound t... (5%)
ET POLICY Suspicious inbound t... (5%)
ET POLICY Suspicious inbound t... (6%)
ET SCAN Potential VNC Scan 590... (9%)
GPL SHELLCODE x86 inc ebx NOOP... (44%)
ET SCAN NMAP OS Detection Prob... (21%)
Global stats
Tests results
85%
8%
8%
full detection
partial detection
no detection
Modules distribution
badTraffic
bruteForce
denialOfService
evasionTechniques
fragmentedPackets
normalUsage
pcapReplay
shellCodes
testRules
bruteForce
evasionTechniques
fragmentedPackets
shellCodes
testRules

Fig. 8 Snort multi thread overview

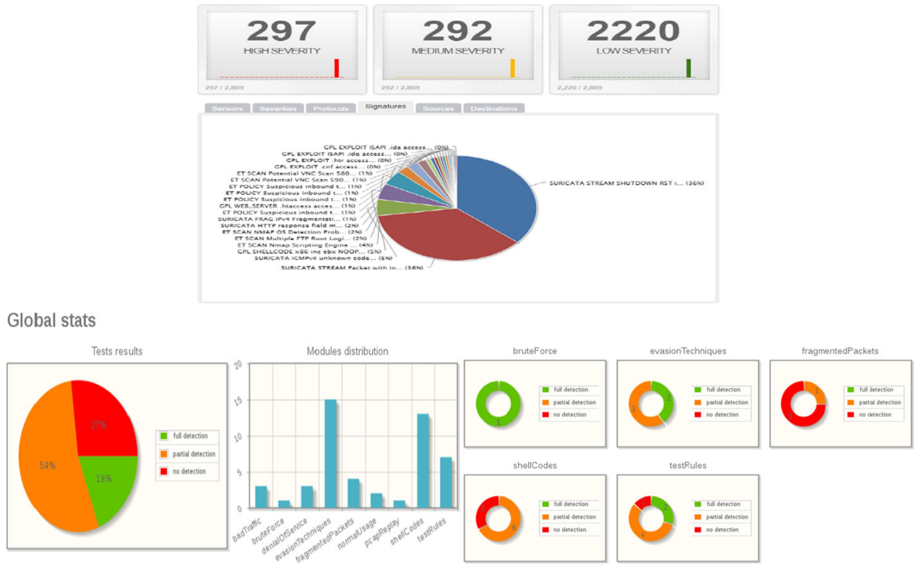
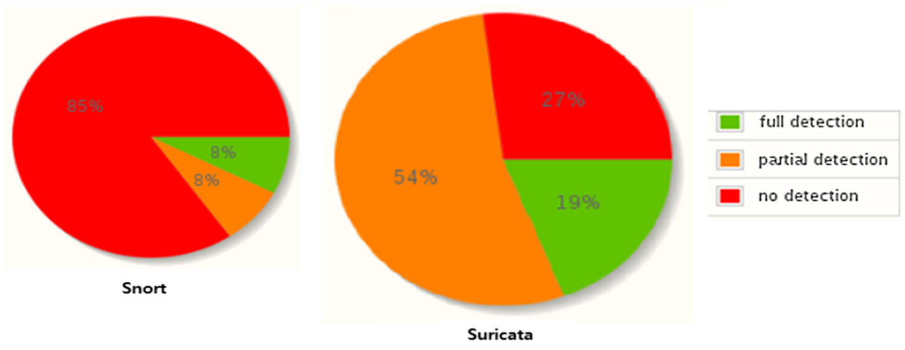

297
HIGH SEVERITY
297 / 2,809
292
MEDIUM SEVERITY
292 / 2,809
2220
LOW SEVERITY
2,220 / 2,809
Sensors
Severities
Protocols
Signatures
Sources
Destinations
GPL EXPLOIT ISAPI .ida access... (0%)
GPL EXPLOIT ISAPI .idq access... (0%)
GPL EXPLOIT .htr access... (0%)
GPL EXPLOIT .cnf access... (0%)
ET SCAN Potential VNC Scan 580... (1%)
ET SCAN Potential VNC Scan 590... (1%)
ET POLICY Suspicious inbound t... (1%)
ET POLICY Suspicious inbound t... (1%)
ET POLICY Suspicious inbound t... (1%)
GPL WEB_SERVER .htaccess acces... (1%)
ET POLICY Suspicious inbound t... (1%)
SURICATA FRAG IPv4 Fragmentati... (1%)
SURICATA HTTP response field m... (2%)
ET SCAN NMAP OS Detection Prob... (2%)
ET SCAN Multiple FTP Root Logi... (2%)
ET SCAN Nmap Scripting Engine ... (4%)
GPL SHELLCODE x86 inc ebx NOOP... (5%)
SURICATA ICMPv4 unknown code... (5%)
SURICATA STREAM Packet with in... (36%)
SURICATA STREAM SHUTDOWN RST i... (36%)
Global stats
Tests results
27%
54%
19%
full detection
partial detection
no detection
Modules distribution
badTraffic
bruteForce
denialOfService
evasionTechniques
fragmentedPackets
normalUsage
pcapReplay
shellCodes
testRules
bruteForce
evasionTechniques
fragmentedPackets
shellCodes
testRules

Fig. 9 Suricata multi thread overview

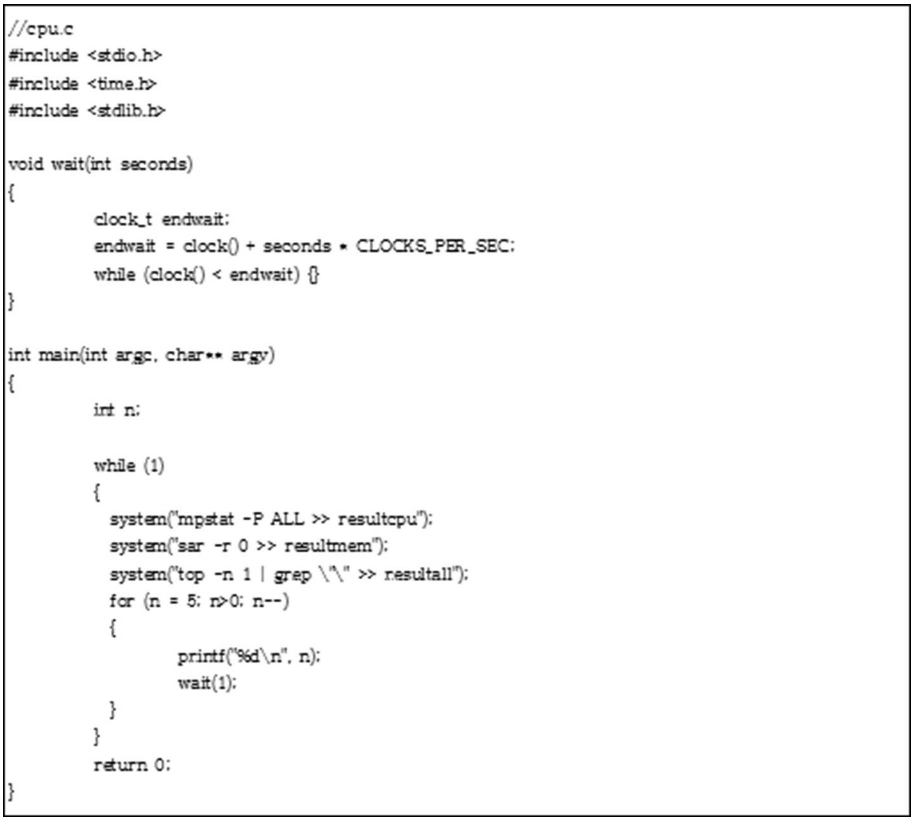
4 Snort Suricata Performance Analysis

In this chapter we have analyzed the performance of Snort and Suricata attacking by DoS and compared the single core and multi core CPU usage.



IDS	High	Medium	Low
Snort	137	166	2
Suricata	297	292	2220

Fig. 10 Snort and suricata multi thread overview



```
//cpu.c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

void wait(int seconds)
{
          clock_t endwait;
          endwait = clock() + seconds * CLOCKS_PER_SEC;
          while (clock() < endwait) {}
}

int main(int argc, char** argv)
{
          int n;

          while (1)
          {
            system("mpstat -P ALL >> resultcpu");
            system("sar -r 0 >> resultmem");
            system("top -n 1 | grep \"\" >> resultall");
            for (n = 5; n>0; n--)
            {
                   printf("%d\n", n);
                   wait(1);
            }
          }
          return 0;
}
```

Fig. 11 Snort and suricata CPU reporting programm source

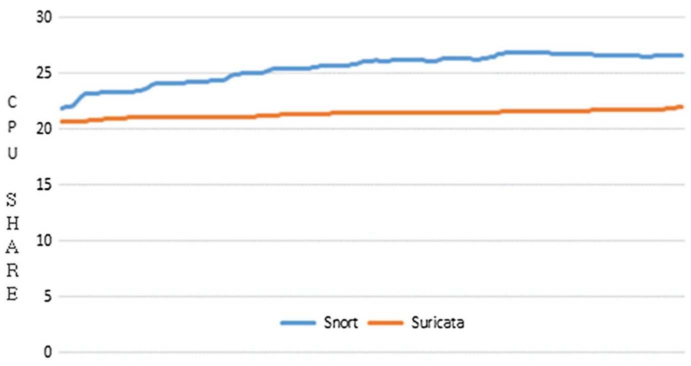


Fig. 12 Snort and suricata CPU performance single CPU

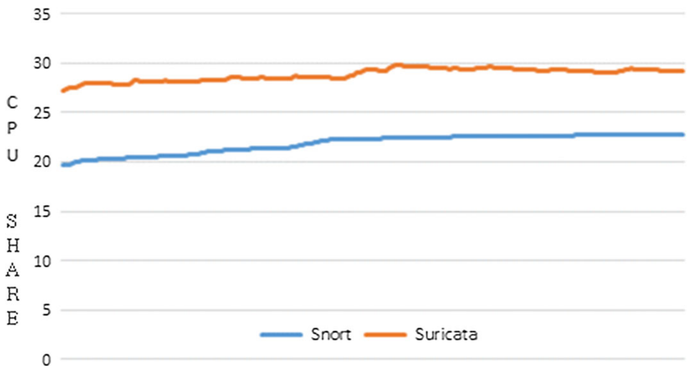


Fig. 13 Snort and suricata CPU performance multi CPU

To record the CPU usage of the OS, we needed a reporting tool. By default, Linux, mpstat shows the CPU usage of each process that takes part. The picture below shows the process to record CPU usage of each program in 5 s (Fig. 11).

The picture below shows Snort and Suricata's CPU shares in single thread. It can clearly be seen that the Snort uses more CPU then the Suricata up to five percent (Fig. 12).

As seen in the picture below Suricata has beneficial of using multi core, which uses more CPU then snort. As the time increases Suricata's CPU share increase as on the other hand Snort's CPU share decreases (Fig. 13).

5 Conclusion

A trend in open source IDS/IPS system was focused on Snort which gained good reputation on its detecting and performance. However, there had been some changes in recent trends. Especially Suricata which had multi-threading technique and lots of other technique such as file Identification & extraction, HTTP normalizer & parser, and automated protocol identification has been added to threat Snorts place in open source IDS/IPS. The result shown in this paper, which we have tested the two IDS mentioned above, have gained results showing each of its own personality. The results show clearly that the Snort had better performance which was low CPU share than Suricata. This result shows that if the user how is trying to implement IDS/IPS in low CPU, Snort is the better solution to choose. However, Suricata which has more advantages than Snort, outstand Snort in its single and multi-core detection performance, and the single core performance. Also Suricata extends the performance if GPU is enabled. By these result we can see that Suricata is slowly outstanding Snort by using multi thread and much of its features.

References

1. Roesch, M. (1999). *Snort: Lightweight intrusion detection for networks*. vol. 229. Santa Clara, CA: Stanford Telecommunications Inc.
2. Zhou, Z., Zhongwen, C., & Tiecheng, Z. (2010). *The study on network intrusion detection system of Snort*. In 2010 2nd International Conference, IEEE.
3. Tjhai, G. C., Papadaki, M., Furnell, S. M., & Clarke, N. L. *Investigating the problem of IDS false alarms: An experimental study using Snort*. In International Information Security Conference.
4. DeLong, R. J., & Los Gatos, C. A. (2001). *Structured exception-handling methods, apparatus, and computer program products*. Sun Microsystems Inc.
5. Chakrabarti S., Chakraborty, M., & Mukhopadhyay, I. *Study of snort-based IDS ICWET 10*. In Proceedings of the International Conference and Workshop on Emerging Trends in Technology, pp. 43–47.
6. Norton, M., & Roelker, D. (2002). *SNORT 2.0: Hi-performance multi-rule inspection engine*. Columbia: Sourcefire Network Security Inc.
7. Day, D. & Burns, B. (2011). *A performance analysis of Snort and Suricata network intrusion detection and prevention engines*. In Fifth International Conference on Digital Society, Gosier, Guadeloupe, pp. 187–192.
8. Garcia-Teodoro, P., et al. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security, 28*(1), 18–28.
9. Caswell, B., Beale, J., & Baker, A. (2007). *Snort IDS and IPS toolkit*. New York: Syngress.
10. Damaye, S. (2011). *Suricata-vs-Snort*. Retrieved from www.aldeid.com/wiki/Suricata-vs-snort, October 2 2011.
11. Watchinski, M. (2011). *Unusual snort performance stats*. Retrieved October 2 2011 from comments.gmane.org/gmane.comp.security.ids.snort.general/30527.
12. Burks, D. (2014). *Security onion: Peel back the layers of your network in minutes*. Pittsburgh, PA: Software Engineering Institute.
13. Deuble, A. (2012). Detecting and preventing web application attacks with security onion. *SANS Institute, 4*(1), 26–33.
14. Bejtlich, R. (2013). *The practice of network security monitoring: understanding incident detection and response*. San Francisco: No Starch Press.

Wonhyung Park received his Ph.D. degree in Department of Information Security from the Kyonggi University, South Korea, in 2009. Now, he is Associate Professor in Department of Cyber Security, Far East University, South Korea. He co-authored more than 50 technical papers in the area of information security. Also, he has been reviewer for International Journal (Computer-Journal) of Oxford Univ. press and IEEE International Conference (ICISA 2011, ICISA 2010).

Seongjin Ahn received his Ph.D. degree in Department of Information Engineering from Sungkyunkwan University, South Korea in 1998. Now, he is Professor in Department of Computer Education, Sungkyunkwan University, South Korea.