ISSN: 2321-7782 (Online)
Volume 3, Issue 2, February 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Smart Snort Enhanced Security in Computer Networks

Ritu Makani[1]
Department of CSE GJUS&T
Hisar - India

Yogesh Chaba[2]
Department of CSE GJUS&T
Hisar - India

Abstract: ***Snort is a Network based intrusion detection system which uses signatures of attacks for pattern matching. It creates rules using rule set language for matching patterns. Snort by default is an intrusion detection system and requires additional plug-ins to work as intrusion prevention system. It can be made to work in three modes. But it will depend upon the positioning of the snort in your system and the cost to be incurred on the security of the overall system that we can make it work in inline mode. There is ample flexibility in manipulating rules as per requirements. Rules were added to Snort to make it Smart Snort. The major drawback of any intrusion detection system is that it detects any threat to the system and logs it but it does not take any action to prevent it except when it is configured to work as an intrusion prevention system. An effort was made to work on snort to customize it by working on rules and making snort to work as an intrusion prevention system thereby increasing its versatility. In addition there was an increase in the packet capture per minute protocol wise and lesser memory consumption. A multi pattern matching algorithm (Aho-corasick) is used to create the Automata for selective option of the rule.***

Keywords: Automata, Pattern Matching, Customized Snort, Aho-Corasick, IDS.

I. INTRODUCTION

The spread of Internet has lead to a boom in various network related activities like banking, E-commerce, Defence Networks, Radar Systems, Social Engineering, Medical and almost every field that can be thought of. System and network security is a key element for all these variety of applications. Encryption, Authentication mechanisms, Intrusion Detection Systems, Security Management can be used to increase the security of the network of computers. Options available can do it but the cost factor does not allow medium and low budget companies to go for it. More effort is required for low cost and effective security systems. It can be done by exploring the options of the presently available options in this field. An intrusion detection system (IDS) is a well known security tool used by companies to prevent loss and harm of data. An open source intrusion detection system is a good option for organizations which do not have the same amount of money as the larger companies. Snort, an open source Network Intrusion Detection System is one such tool which can be worked with to optimize for its efficiency and speed. Working and efficiency of and IDS will depend upon, where an IDS has been placed in the system. Based on the deployment it is classified as:

Host-based monitoring: A host-based IDS (HIDS) is deployed on devices that have other primary functions such as Web servers, database servers and other host devices. A host-based IDS provides information such as user authentication, file modifications/deletions and other host-based information, thus designated as secondary protection to devices on the network. Examples of HIDS products are EMERALD, NFR etc. The main advantages of Host-based Intrusion Detection Systems is that HIDS can collect much more detailed information regarding exactly what occurs during the course of an attack and due to increased granularity of tracking events in the monitored system, recovery from a successful intrusion incident is usually more complete. Since the attack affects the monitored host, HIDS detects unknown attacks more than the Network-based IDS with few false positives. Disadvantage of HIDS is that due to network heterogeneity and the profusion of operating systems, no single host-based IDS can translate all operating systems, network applications, and file systems. In addition, in the absence of

something like a corporate key, no IDS can decipher encrypted information. Host-based IDS usually rely heavily or completely on an audit record of activity that is created by a system or application. This audit record varies widely in quality and quantity between different systems and applications, thus dramatically affecting IDS effectiveness. It is expensive as HIDS are installed on the system being monitored. On very large networks this is both very expensive and difficult to manage. HIDS detects only after the intruder has reached the monitored host system, not before, as can network-based IDS. Moreover it overloads host CPU such that it interfere with normal host operations.

Network-based monitoring: Network-based IDS (NIDS) is to monitor the traffic of that network. So NIDS can observe all communication between a network attacker and the victim system, evolving many of the problems associated with log monitoring. Typical Network-based IDS are Microsoft Network Monitor, Cisco Secure IDS (formerly NetRanger), Snort etc. Main Advantages of network-based intrusion detection system is ease of deployment due to passive nature and hence few performance or compatibility issues in the monitored environment. Further these are cost effective as strategically placed sensors can be used to monitor a large organizational environment. Moreover the variety of malicious activities able to be detected through the analysis of network traffic is wider than HIDS. Since NIDS sensors run on a host separate from the target, they are more impervious to tampering. It detects all attempts, even failed ones where as HIDS detects only successful attacks because unsuccessful attacks do not affect the monitored host directly. But NIDS are susceptible to packet spoofing, which tricks the IDS into thinking that packets have come from an incorrect location or packet fragmentation attacks. Moreover NIDSs often cannot decipher the packets they capture. In addition, in the absence of something like a corporate key, no IDS can decipher encrypted information. These may get failed when the monitored network is heavily loaded. Another problem can be that NIDS can process packets on low-speed networks (10Mbps), few claim to be able to keep up and miss no information at 100Mbps or higher. Providing complete coverage can be costlier and problem is worsened if working in switched networks. NIDS focus is on detecting attacks from outside, rather than attempting to detect insider abuse and violations of local security policy.

Host network monitoring: Host Network Monitoring combines network monitoring with host-based probes. It resolves many of the problems associated with promiscuous network monitoring, while maintaining the ability to observe the entire communication between victim and attacker. Like all host-based approaches, however, this approach implies a performance impact on every monitored system, requires additional support to correlate events on multiple hosts, and is subject to subversion when the host is compromised. Sometimes this hybrid intrusion detection system is considered as a subtype of network-based intrusion detection system because it relies primarily upon the network traffic analysis for detection. Example of hybrid IDS is prelude.

Target based monitoring: These use scanning techniques to form an image of what systems exist in the protected network, including such details as host operating system, active services, and possible vulnerabilities. Using this knowledge, a probe can reconstruct network traffic in the same fashion, as would be the case on the receiver system, preventing attackers from injecting or obscuring attacks. It allows IDS to automatically differentiate attacks that are a threat to the targeted system, from those that target vulnerabilities not present – thus refining generated alerts. Discussion above provides the scenario, advantages, disadvantages, competencies expected from IDS.

II. PREVIOUS WORK DONE

Toby Kohlenberg; Raven Elder; James C. Foster; Matt Jonkman; Rafel Marty; Mike Poor "*Snort IDS and IPS Toolkit*" Jay Beale's Open Source Security Series provided the basics of the snort detection engine and how to work with its rule sets and its various working modes [1]. Coint CJ.; Stanford S; and McAlemey J worked towards faster string matching for intrusion detection for exceeding the speed of Snort[3] and Aho A.V.; Corasick M.J.(1975) worked on efficient string matching as an aid to bibliographic search"[4]. Zhou Zhimin *et al* performed the study on network intrusion detection system of Snort"[5]. Chintan C. Kacha *et al* worked for improvement in Snort intrusion detection system using modified pattern matching technique[6].

Tongaonkar A *et al* threw some light on fast packet classification for Snort by native compilation of rules"[9]. Jiqiang Zhai *et al* worked on network intrusion prevention system based on Snort. Deepak Dembla et.al have discussed [10] about modelling and analysis of intelligent AODV routing protocol based on request retransmission strategy in MANETS. Yogesh Chaba et.al in their work [11] have discussed about performance analysis of disable IP broadcast technique for prevention of flooding-based DDoS attack in MANETS. Further Yudhvir Singh et.al has described about information theory tests based performance evaluation of cryptographic technique [12]. But none of the above talked at length on how to make snort a versatile and customizable IDS/IPS tool and which has been talked about in this work.

III. PROPOSED WORK

Based on the above discussions we conclude that there is a need of a system that overcomes the drawbacks of speed, cost effectiveness and versatility. So our aim was to find solution to these issues and the present paper is a step in that direction. We opted to work with an open source IDS snort and tried to make use of its flexibility to work with its rule set and received good results with regard to traffic monitoring while maintaining high throughput. Further working with the scheme of things i.e automata provides to make it to work for just not detecting but preventing the attack also, by working as intrusion prevention system also.

Introduction to Snort: Snort is open source intrusion detection software which runs on Windows or Linux operating systems. Being free and having complete set of capabilities and the possibility to be installed on different machine and operating systems made Snort a popular IDS in computer networks. Snort in complete version is a kind of Network Intrusion Detection System which follows a unixy configuration philosophy. Its configuration is plaintext but it is powerful & complex. Snort configuration consists of Global configuration file snort.conf which are called Gold Standard rules and optional *rules* files which are framed by users and can be shared on a public platform. Snort bleeding edge rules are these kind of user framed rules which are untested but floated for use and for comments. So this provides ample opportunity for users to formulate rules for their needs. This software is configurable in three modes: sniffer mode, packet recording mode and ID system. Sniffer mode only detects the content of the transmitted packet and logger mode stores the data in a file and only intrusion detection mode analyze data based on rules. Snort checks network traffic based on a characteristic database of invading programs. For example someone can adjust Snort by a rule to make a warning message or to take proper action whenever an access in a defined protocol from/to a specific port and from/to specific destination with a content containing a specific string happens.

Description of a Snort Rule: 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 rule set. 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.

3.1 working with rules

Following considerations have been kept in mind while working with Snort rules to maximize efficiency and speed.

- Using TCP Flag Tests to speed up Content Rules: As the content rules are tested at last we can take advantage of this fact by using other faster rule options that can detect whether or not the content needs to be checked at all. For instance, most of the time when data is sent from client to server after a TCP session is established, the PSH and ACK TCP flags are set on the packet containing the data. This fact can be taken advantage of by rules that need to test

payload content coming from the client to the server with a simple TCP flag test that is far less computationally expensive than the pattern match algorithm. The basic idea is that if the PSH and ACK flags aren't set, there's no need to test the packet payload for the given rule. If the flags are set, the additional computing power required to perform the test is negligible.

- Content Rules are Case Sensitive: As content rules are case sensitive and that many programs typically use uppercase letters to indicate commands. FTP is a good example of this.
- Changing the order of the rule: The last rule test that is done (when necessary) is always the content rule option. We can change this order to check the required option first.
- Snort explains well-known and common vulnerability exploitation attempts, violations of security policy and conditions under which a network packet(s) might be anomalous.

Work was done with some rules modifiers commands related to protocols like tcp, udp, icmp, ipv4. Wide difference was observed in the alerts generated with standard rule set and modified rule set.

3.2 WORKING WITH AUTOMATA:

By working with rule tree structure (automata) and using the inline mode of snort it was made smart Snort and it worked like an IPS with faster packet processing maintaining high throughput. Snort performs multi-pattern matching using Aho-Corasick algorithm for intrusion detection. The library constructs the automata for the entire rule files based on the protocol type and rule tree structure (automata) will be constructed for both the header and the options portion occupying more memory. More states and transitions will take more time to do the transition.

Aho-Corasick Algorithm: It locates all the occurrence of set of patterns in a text of string. It consists of constructing a finite state pattern matching automata from the patterns and then using the pattern matching automata to process the text string in a single pass. Packet data patterns are matched as per rules and this is done by pattern matching algorithms, which works as follows:

Preprocessing Phase:

Step 1: Automata Construction: Construct finite state automata for the set of predefined patterns (or pattern tree) which are supposed to be found in the text string. The states will be numbered by their names and transitions between the defined states would be represented by the characters existing in the particular pattern.

Step 2: Failure Function: Failure function can be defined as the longest suffix of the string that is also the prefix of some node. The goal of the failure function is to allow the algorithm not to scan any character more than once. After constructing automata, failure function of each node is calculated and its corresponding transitions are also required to be mentioned, so the constructed automata would be called as "Automata with failure links".

Step 3: *Output Function*: Lastly in the automata output function for final states has to be calculated for recognizing the pattern string which may be found in the text string. And the resulting automata would be called as “Automata with Output Functions” Output function gives the set of patterns recognized when entering final state.

Searching Phase By using the Aho Corasick Searching Algorithm search the text using the pre constructed Finite State Automata for the set of predefined patterns.The searching phase of aho corasick is straightforward while scanning the text it walk through automata if any transition found, it get transition, otherwise check the failure function. Aho- Corasick is applied for Digital Forensics. In which string searches are designed to search every byte of the digital evidence, at the physical level, to locate specific text strings of interest to the investigation. Given the nature of the data sets typically encountered.

Although the Aho-corasick algorithm used by snort outperforms others like Booyer Moore and Bloom filter but occupies large memory space in the system for pattern matching. It was tried to work simultaneously on rule tree structure (automata) of Snort

and on the specific processing of rule options to reduce this memory consumption as main target was to reduce memory consumption. By working with modified automata and more number of rules Snort works as Smart Snort. Following features are achieved which make it Smart Snort:

- In case of smart Snort the finite automata construction for the header portion is completely avoided and the automata is constructed for the options portion of the rule alone. This is further reduced by restricting the automata construction only for the content option/port no of the predefined rule set. Figure 1 shows the same.
- Smart Snort does both detection and prevention tasks with reformulated automata and more number of modified rules, when it is working in the inline mode. Also in this approach, integration of both the misuse and the anomaly detection happens. For implementing the misuse based detection system in smart Snort the default Snort rules were modified.
- Smart Snort reduces the number of searches considerably by confining the searches to the protocol to which the packet is associated to because initial pre-processing is done by grouping the rules based on the protocols and storing them separately in four different files for TCP,UDP,ICMP and IP respectively. Otherwise packet will be compared against all the rules irrespective of its protocol.
- As a packet is trapped in smart Snort, it extracts the protocol field from the packet‘s header. Packets are then redirected to the file that contains the rules pertaining to the protocol to which the packet is associated to.
- Smart Snort will process all the packets which it receives without much packet loss preserving high throughput, and response time.

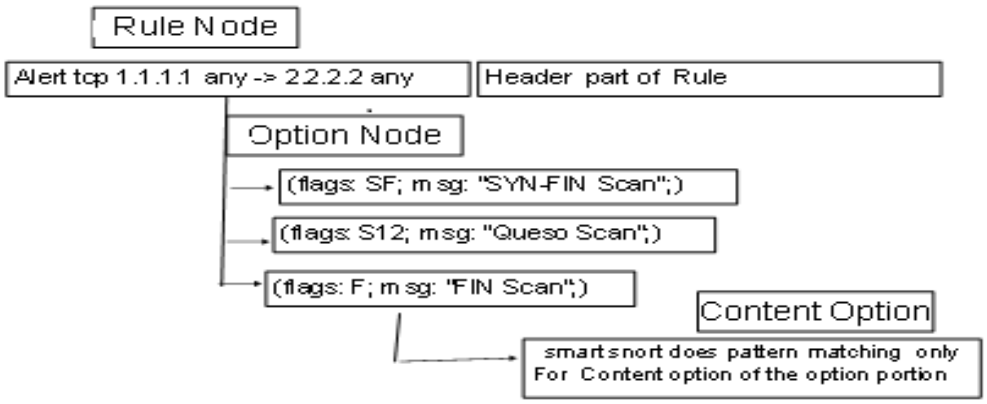


Figure 1 Content option of the rule

IV. RESULTS

Network Performance Evaluation of Security Protocols

Before discussing the results let us discuss some parameters on which the performance evaluation of the implemented scheme/mode on which work has been done can be checked for any change in the existing scheme of things. Alerts received, packets captured/minute, packet processing rate/min, execution time are some of these parameters. Further rule modifications in respect of the above mentioned parameters are also one of them.

(a)

Comparativr memory consumption for snort and smart snort

Smart Snort	Snort
17.8	25.8
21.4	30.54
24	35
21.1	29.1
24	35
23.3	30.3
22.8	27.4
22.8	27.4
19.2	29.2
28.4	31.4
26.4	35.4

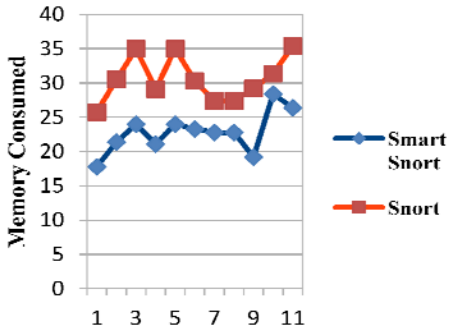


Figure in section a) shows the comparison of memory consumption of snort and smart snort while working in the inline mode with >20000 rules

(b)

% age of Processor Tme Comparison B/T Snort & Smart Snort

Smart Snort	Snort
38.53	43.3
36.53	46.4
50.1	55.3
36.2	42.14
41.3	52
33.5	42.7
38.9	53.8
38	45
40	49.8

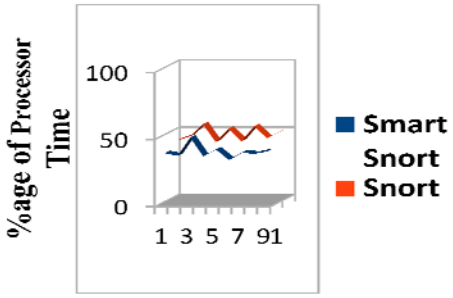
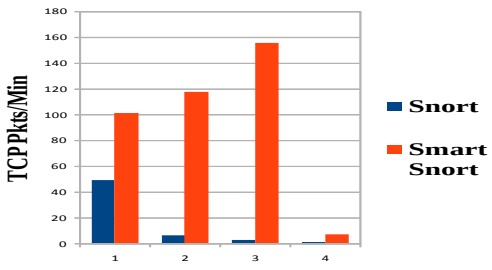


Figure in section b) shows the comparison of the %age of processor time used by Snort and Smart Snort. Smart Snort uses less processor time than snort with standard configuration.

(c)

TCP Pkts received /Min

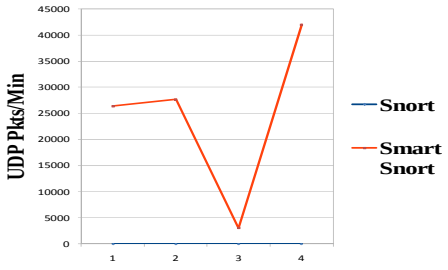
	Snort	Smart Snort
	49.62	101.24
	6.65	117.89
TCP	3.04	155.78
	1.75	7.7



(d)

Udp packets received/min

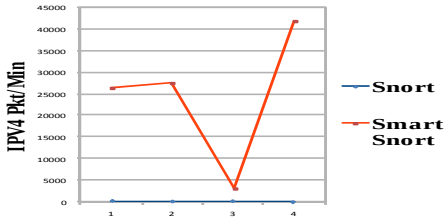
	Snort	Smart Snort
	7.5	26375
	12.91	27647
UDP	14.81	3053.75
	3.42	41961



(e) IPV4 pkts recieived/min

I

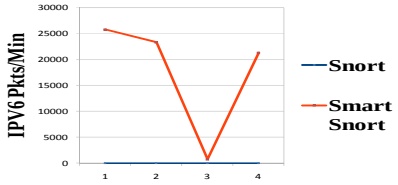
	Snort	Smart Snort
	233.66	26439
	139.71	27668
IPV4	179.16	3066
	20.06	41977.22



(f)

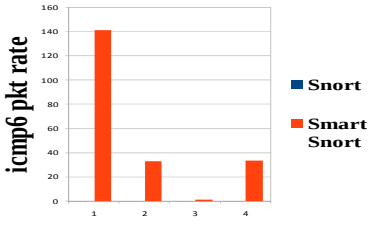
Ipv6 pkts recieved /min

	Snort	Smart Snort
	0.07	25803.34
	0.125	23301.92
IPV6	0.2	724.12
	0.12	21320.03



(g) ICMP Pkts/Min Received

	Snort	Smart Snort
	0.06	140.99
	0.1	33.28
ICMP6	0.13	1.2
	0.11	33.58



Figures in section c, d, e, f, g sections show the comparative tcp, udp, ipv4, ipv6 and ICMP6 Packets received per minute between Snort and Smart Snort.

(h)

Day of Week	Std. Snort Rules	smart Rules
1st Day	14	52
2nd Day	10	83
3rd Day	11	77
4th Day	9	67
5th Day	13	88
6th Day	19	79

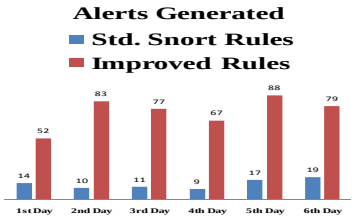


Figure in section h shows the comparative results of the snort and smart snort with standard and modified rules

V. CONCLUSION

Working with an open source IDS like Snort gives much of leverage to the strategic security design teams. Rule options give ample scope for modification and manipulation as per the need of the security requirements of a concern. Further variable working modes of the Snort IDS (passive, inline) makes it more versatile because it can also work as an intrusion prevention system, which otherwise it is not in its default mode. Results of protocol wise packets received per minute in the present work show that throughput is maintained irrespective of its mode with faster packet processing and decreased memory requirements when used with improved rule options. In default snort automata does matches against all the rules whereas smart snort does it according to the new automata which makes it to do pattern matches against only specific part of the option portion of the rule. Automata are basically the guiding program which IDS has to follow for pattern matching. This has been achieved by working on the automata of the multi pattern matching algorithm, which it uses for pattern matching and made snort more secure and versatile.

References

1. Toby Kohlenberg, Raven Elder, James C. Foster, Matt Jonkman; Rafel Marty; Mike Poor “Snort IDS and IPS Toolkit” Jay Beale’s Open Source Security Series.
2. snort homepage http://www.snort.org/.
3. Coint CJ.; Stanford S; and McAlemey J. (2001): Towards Faster String Matching for Intrusion Detection for Exceeding the Speed of Snort, pp.367-373.
4. Aho A.V.; Corasick M.J.(1975): Efficient String Matching: An Aid to Bibliiographic Search, Bell Laboratories, Communication of the ACM, pp.333-340.
5. 2010 International Conference on Networking and Digital Society The Study on Network Intrusion Detection System of Snort Zhou Zhimin, Chen Zhongwen , Zhou Ti echeng, Guan Xiaohui Department of Computer Science Zhejiang Water Conservancy And Hydropoeer College Hangzhou, China.
6. Chintan C. Kacha et al “Improved Snort Intrusion Detection System Using Modified Pattern Matching Technique” IJTEAE(ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 7, July 2013.
7. Roesch M. (1999): Snort;Lightweight intrusion detection for networks, In Proceedings of the 1999 USENIX LISA Systems Administration Conference, pp.229-238.
8. Comparison of Different Intrusion Detection And Prevention Systems, Abhishek Chauhan.
9. Tongaonkar A; Vasudevan S; Sekar R(2008): Fast Packet Classification for Snort by Native Compilation of Rules, Stony Brook University, 22nd Large Installation System Administration Conference, pp.159-165.
10. http://sourceforge.net/projects/multifast/.
11. 2011 The 6th International Forum on Strategic Technology Researh on Network Intrusion Prevention System Based on Snort Jiqiang Zhai, Yining Xie Computer Science & Technology College,Harbin University of Science and Technology Harbin,China.
12. Deepak Dembla, Yogesh Chaba, “Modeling and Analysis of an intelligent AODV Routing Protocol based on Route Request Retransmission Strategy in MANETs” International Journal of Computer Applications (0975-8887), Vol. 30, Issue 11, pp. 6-13 (2011).
13. Yogesh Chaba, Yudhvir Singh, Preeti Aneja, “Performance Analysis of Disable IP Broadcast Technique for Prevention of Flooding-Based DDoS Attack in MANET” Journal of Networks, Vol 4, Issue 3, pp. 178-183 (2009).
14. Yudhvir Singh, Yogesh Chaba, “Information theory tests based performance evaluation of cryptographic techniques” International Journal of Information Technology & Knowledge Management, Vol. 1, Issue 2, pp.475-483 (2008).
15. Roesch Martin, ”snort manual” (2003).

