International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 05 | May-2016 www.irjet.net p-ISSN: 2395-0072

A Review of Network Intrusion Detection Systems

Mala Dutta
Assistant Professor, Department of Computer Engineering, IET DAVV, Indore

---

Abstract - *This paper talks about analysis of Network Intrusion Detection Systems, which inspect the incoming packets for known intrusion-related signatures or anomalies, related to Internet protocols. Based upon a set of signatures and rules, the detection system is able to find and log suspicious activity and generate alerts. The paper includes comparison of the algorithms used in such systems and also presents some of the rule sets used for testing of network intrusion detection systems.*

Keywords : Intrusion detection, network packet inspection, attack signature

1. Introduction

We start by discussing some of the string matching algorithms [4] used by network intrusion detection systems:

A. Naïve String Matching Algorithm

The time required by Naïve string matching algorithm is 0 for preprocessing time.

□ □ □ □ □ □ □ □ x (m-n+1□ □ □ □for□ pattern matching time

where 'm' is the number of characters in the text

The time turns out to be so because in an 'n' length pattern, we require to match pattern for those 'n' characters and this procedure has to be followed m-n+1 times.

B. Rabin-Karp Algorithm

The time required by Rabin-Karp algorithm is

□ □ □ □ □ □ □ □□□□□□n□ □ □for preprocessing time

where 'n' is the number of characters in the pattern

□ □ □ □ □ □ □ □□□□□□x □m-n+1□ □ for pattern matching time

where 'm' is the number of characters in the text.

C. Normal Finite Automaton Algorithm

The time required by finite automaton algorithm is:

□ □ □ □ □□□□n x |E|) □ □for preprocessing time,

where 'n' is the number of characters in the pattern

□ □ □ □ □□□□□n□ □ □for pattern matching time

where 'm' is the number of characters in the text.

'E' is the alphabet set of the finite automaton and hence |E| = the number of alphabets in the finite automaton.

D. Knuth-Morris-Pratt Algorithm

The time required by *Knuth-Morris-Pratt* algorithm is

□ □ □ □ □ □ □□□□□n□ □ □for preprocessing time

where 'n' is the number of characters in the pattern.

□ □ □ □ □ □ □ □□□□ m):for pattern matching

where 'm' is the number of characters in the text.

E. Boyer-Moore Algorithm

The time required by the Boyer Moore algorithm is

□ □ □ □ □ □ □ □ □(n) □for preprocessing time

where 'n' is the number of characters in the alphabet set

□ □ □ □ □ □ □ □ □ □n/n) for pattern matching (best case).

where 'm' is the number of characters in the text.

Implementation of network security can be categorized into three modes: defense (firewalls), deterrence (cyber laws) and detection. The third one is least commonly implemented on network systems [3]. Signature Detection

Engines are a part of Intrusion Detection Systems, which are used to carry out detection. Signature Detection Engines rely on signatures to detect the packet anomalies and malicious content in the payload. Signature is the pattern you look inside a packet. A signature may be used to detect one or multiple types of attacks [1]. For example, the presence of SYN and FIN Flags in the same packet may indicate an intruder activity. Signatures may be present in different parts of a data packet depending upon the nature of the attack. For example, we can find signatures in the IP header, transport layer header (TCP or UDP header) and/or application layer header or payload. Any efficient engine should be able to scan all of the above.

2. Analysis of the algorithms

Both the KMP and BM algorithms begin by computing a shift function. Following the precomputation of shift function, the actual match is carried out. The complexity of the algorithms is given by the number of comparisons required in the matching stage (excluding the precomputation stage). The KMP algorithm has better worst-case time complexity (O (m+n); O (mn) for Boyer Moore) where as the BM algorithm has better average-case time complexity [5]. We use the above facts to our advantage in the engine by allowing the administrator to switch between the two algorithms depending upon the vulnerability of the network. When a network is more prone to attacks, i.e. more number of malicious packets are entering the network, the hit ratio of the engine is high (best case) and the administrator can select the BM mode and correspondingly, when the hit ratio is low (worst case), the administrator can switch to KMP mode. The compromise on the hit time is over-compensated by the faster processing of a bulk of the packets not containing any malicious code.

3. Testing for NIDS

In this section we provide some of the rules [2], which are used to analyze and test the performance of network intrusion detection systems.

A. Policy Rules

Signature:

Protocol: tcp Source IP: Any
Destination IP: Any Source Port: Any
Destination Port: 5050 Flags: Any
Payload: "<Ymsg Command="

Description: A policy rule gets activated when an attempt is made to violate the policies of an organization.

B. ICMP Rules

Signature:

Protocol: icmp Source IP: any
Destination IP: Any Type: 8
Code: 0Payload: “ISSPNGRQ”

Description: This event is generated when an ICMP echo request is made from a host running the Internet Security Scanner tool with intent to gather information.

C. Scan Rules

Signature:

Protocol: tcp Source IP: Any Destination IP: Any Source Port: Any Destination Port: 80 Flags: S FPayload: Any

Description: This event indicates that an attempt has been made to scan a host. Scanners are used to determine which ports a host may be listening on, whether or not the ports are filtered by a firewall and if the host is vulnerable to a particular exploit.

D. Shellcode Rules

Signature:

Protocol: ipSource IP: AnyDestination IP: AnySource Port: AnyDestination Port: all shell portsFlags: AnyPayload: “|EB|AX|8B D8 8B|3|8B|{|04 03 F7 8B|K|08 03 CE|3|C0 B0 08|@@@@|03|"

Description: This event indicates that shellcode has been detected in network traffic. A shell code starts a command shell from which the attacker can control the compromised machine.

4. Conclusion

We have analyzed the algorithms used in network intrusion detection systems and the various tests than can be done on them. But false positives still remains the big concern in such systems. To avoid false alarms, we have to modify and tune different default signatures and in some cases we may need to disable some of the signatures to avoid false alarms. Using automated network attack signature generation models can reduce some of the false positives.

References

[1] Jia Ni1, Chuang Lin1, Zhen Chen1, 2 and Peter Ungsunan1, A Fast Multi-pattern Matching Algorithm for Deep Packet Inspection on a Network Processor

[2] Attack Detection and Signature Generation, Sixth Framework Program, European Networks of Affined Honeypots.

[3] Mark Rhodes-Ously, Roberta Bragg, Keith Strassberg: "The Complete Reference, Network Security".

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: "Introduction to Algorithms", Second Edition.

[5] Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran: "Computer Algorithms, C++".