ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 5, May 2013

Novel Algorithm for Intrusion Detection System

Lata[1], Kashyap Indu[2]

Dept. of CSE, FET, MRIU[1]

Faridabad, Haryana, India[2]

Abstract: Intrusion detection system is device or software applications that monitor network or system activities for malicious activities or policy violation. Two types of Intrusion detection systems are network based and host based. This paper is only discussed about network based intrusion system. Snort and Sax2 are network based intrusion detection system. These systems monitor the network and capture packets in promiscuous mode, analyze these packets and give report. Three methodologies are used for detect intrusion on the Network, signature based, anomaly based and stateful protocol analysis. This paper is based on the signature based intrusion detection system methodology. Intrusion can be possible on the header part or payload part .Different pattern matching algorithms are used for detection intrusion. Brute force and Knuth-Morris-Pratt are two single keyword pattern matching algorithms detect the payload part intrusion. String matching consists in finding one or more occurrences of a pattern in a text (input) if Pattern is present in the text send intrusion alarm. False alarm is very high in intrusion detection. I proposed a string matching algorithm to reduce false alarming percentage.

Keywords: Intrusion detection system (IDS), network behavior analysis system (NBAS), network based intrusion detection system (NIDS), TCP, UDP, intruders, attacks, signature, stateful, anomaly, false alarm.

I. INTRODUCTION

Network based intrusion detection system monitor network activities. Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies[9]. Intrusion detection systems (IDS) are primarily focused on identifying possible incidents, logging information about them, and reporting them to security administrators. IDSs typically record information related to observed events, notify security administrators of important observed events, and produce reports. Due to the growth of network environment complexity, the necessity of packet payload inspection at application layer is increased. String matching, which is critical to network intrusions detection systems, inspects payloads and detects malicious network attacks using a set of rules. I am proposing a string matching algorithm which will reduce the false alarm percentage.

II. INTRUSION DETECTION SYSTEM

Intrusion Detection Systems help information systems prepare for, and deal with attacks. They accomplish this by collecting information from a variety of systems and network sources, and then analyzing the information for possible security problems. Intrusion detection system

- Monitoring and analysis of user and system activity.
- Auditing of system configurations and vulnerabilities.
- Assessing the integrity of critical system and data files.
- Statistical analysis of activity patterns based on the matching to known attacks.
- Abnormal activity analysis , Operating system audit[1].

III. METHODOLOGIES OF IDS

Intrusion detection system uses many methodologies to detect incidents. Most IDS technologies use multiple detection methodologies, either separately or integrated, to provide more broad and accurate detection.

A. Signature based detection

A signature is a pattern that corresponds to a known threat. Signature based detection is process of comparing signatures against observed events to identify possible incidents. Signature based detection is very effective at detecting known threats but largely ineffective at detecting previously unknown threats. Example: An email with the subject of free pictures and attachment filename of freepics.exe, these characteristics are known form of malware. If attackers modify the file name freepics.exe to freepics1.exe, signature based detection will not able to detect this malware.

Limitations:

It cannot detect previously unknown threats.[2]

B. Anomaly based detection

Anomaly based detection is a process of comparing definitions of what activities is considered normal against observed events to identify significant deviations. An IDS using anomaly based detection has profiles that represent the

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 5, May 2013

normal behavior of such things as users, hosts network connections or applications. The profile is developed by monitoring and characteristics of typical activities, number of email send by user, number of failed login attempts for a host and the level of processor usage for a host over a period of time. Anomaly based detection is very effective at detecting previously unknown threats.
Limitations:
Building profile is very challenging [2].

C. Stateful protocol analysis
Stateful protocol analysis is a process of comparing predetermined profiles of generally accepted definitions of benign protocol activities for each protocol state against observed events to identify deviations. Stateful protocol analysis relies on vender developed universal profiles that specify how particular protocol should and should not be used. The stateful in stateful protocol analysis means that the IDS is capable of understanding and tracking the state of network, transport and application protocols that have a notion of state.
Limitations: It is limited to examining a single request or response. Many attacks cannot be detected by looking at one request - the attack may involve a series of requests [2].

IV. TECHNIQUES of INTRUSION DETECTION SYSTEM

IDS use several techniques, which involve the IDS stopping the attack itself, changing the security environment (e.g., reconfiguring a firewall), or changing the attack's content. The types of IDS technologies are differentiated primarily by the types of events that they monitor and the ways in which they are deployed.
A. Network behavior analysis (NBA), which examines network traffic to identify threats that generate unusual traffic flows, such as distributed denial of service (DDoS) attacks, certain forms of malware, and policy violations (e.g., a client system providing network services to other systems). Behavior-based analysis learns the normal behavior of traffic and systems and then continually examines them for potentially harmful anomalies and for behavior that frequently accompanies incidents. This approach recognizes attacks based on what they do, rather than whether their code matches strings used in a specific past incident. "It stops traffic that is not malicious on its face but that will do malicious things," said Allan Paller [8].
D. Wireless
This technique monitors wireless network traffic and analyzes it to identify suspicious activity involving the wireless networking protocols themselves [3].
B. Host-based
It can analyze activities on the host it monitors at a high level of detail, it can often determine which processes and/or users are involved in malicious activities. Host-based IDSs can detect attacks undetectable to the network-based IDS and can gauge attack effects quite accurately[2].
C. Network-based
It examines or monitors an entire, large network with only a few well-situated nodes or devices and imposes little overhead on network devices and analyzes the network and application protocol activity to identify suspicious activity. Network-based IDSs are mostly passive devices that monitor ongoing network activity without adding significant overhead or interfering with network operation. They are easy to secure against attack and may even be undetectable to attackers; they also require little effort to install and use on existing networks [2].
Sensors can be deployed in one of two modes. Inline mode and Passive mode **Inline mode**: - An inline sensor is deployed so that the network traffic it is monitoring must pass through it, much like the traffic flow associated with a firewall. In fact, some inline sensors are hybrid firewall/IDS devices, while others are simply IDSs. The primary motivation for deploying IDS sensors inline is to enable them to stop attacks by blocking network traffic. Inline sensor systems are intrusion detection and prevention systems [2]. **Passive mode**: A passive sensor is deployed so that it monitors a copy of the actual network traffic; no traffic actually passes through the sensor. Passive sensors are typically deployed so that they can monitor key network locations, such as the divisions between networks, and key network segments, such as activity on a demilitarized zone (DMZ) subnet [2]. Most techniques for having a sensor prevent intrusions require that the sensor be deployed in inline mode, not passive. Because passive techniques monitor a copy of the traffic, they typically provide no reliable way for a sensor to stop the traffic from reaching its destination. In some cases, a passive sensor can place packets onto a network to attempt to disrupt a connection, but such methods are generally less effective than inline methods. Generally, organizations should deploy sensors inline if prevention methods will be used and passive if they will not.

V. NETWORK BASED SYSTEM ARCHITECTURE

Passive sensors can monitor traffic through various methods-

- Spanning Port. Many switches have a spanning port, which is a port that can see all network traffic going through the switch. Connecting a sensor to a spanning port can allow it to monitor traffic going to and from many hosts [2].
- Network Tap. A network tap is a direct connection between a sensor and the physical network media itself, such as a fiber optic cable. The tap provides the sensor with a copy of all network traffic being carried by the media [2].
- IDS Load Balancer. A load balancer can receive copies of network traffic from one or more spanning ports or network taps and aggregate traffic from different networks

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 5, May 2013

(e.g., reassemble a session that was split between two networks). The load balancer then distributes copies of the traffic to one or more listening devices, including IDS sensors, based on a set of rules configured by an administrator. The rules tell the load balancer which types of traffic to provide to each listening device [2].

Components of IDS
Fig 1 is show passive intrusion detection system. The typical components in an IDS solution are as follows:

- Sensor or Agent. Sensors and agents monitor and analyze activity. The term sensor is typically used for IDSs that monitor networks, including network-based, wireless, and network behavior analysis technologies.
- Management Server. A management server is a centralized device that receives information from the sensors or agents and manages them. Some management servers perform analysis on the event information that the sensors or agents provide and can identify events that the individual sensors or agents cannot. Matching event information from multiple sensors or agents, such as finding events triggered by the same IP address, is known as correlation. Management servers are available as both appliance and software-only products.
- Database Server. A database server is a repository for event information recorded by sensors, agents, and/or management servers. Many IDPSs provide support for database servers.
- Console. A console is a program that provides an interface for the IDPS's users and administrators. Console software is typically installed onto standard desktop or laptop computers. Some consoles are used for IDPS administration only, such as configuring sensors or agents and applying software updates, while other consoles are used strictly for monitor [3].

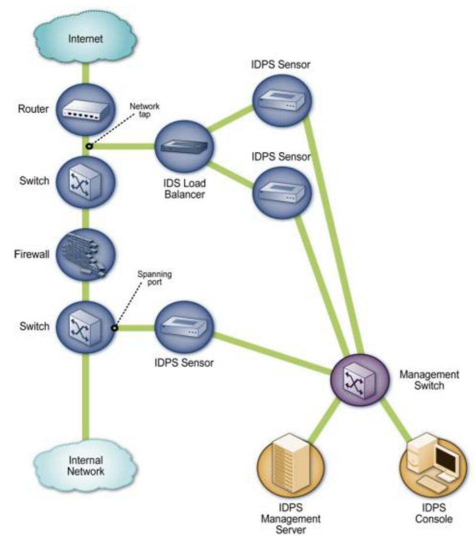

Fig1: Passive Intrusion Detection System

VI. SAX2 NETWORK BASED IDS

SAX2 is a network based IDS. Sax2 is a professional intrusion detection and prevention system that performs real-time packet capturing, 24/7 network monitoring, advanced protocol analyzing and automatic expert detection. By giving insights into all of your network's operations, Sax2 makes it easy to isolate and solve network problems, identify network bottleneck and bandwidth use, detect network vulnerabilities and discovered the network whether there is a breach of security strategy and the signs of being attacked in the network of hazard, and then intercept and stop before their invasion. Network administrators can directly monitor http requests, email messages, ftp transfers, as well as real-time activities and message details for the two popular instant messengers [7]. Sax2 is designed to be used by both IT professionals and novice users. Problems are clearly identified, and solutions are suggested in understandable terms

A. Features of SAX 2

1) *Network based SAX2*: is a network-based IDS. It collects, filters, and analyzes traffic that passes through a specific network location where it placed. Sax2 does not use or require installation of client software on each individual, networked computer.
2) *Intrusion detection and prevention:* Detects variety of complex attacks in the network, including pre-attack detection, password guessing, denial of service attacks (DoS/DDoS) etc. Sax2 will in itiatively stop the dangerous behavior to prevent the whole network.
3) *Traffic analysis*, With its real-time display and statistical traffic analysis of whole network, you may find network resource abuse, worms, denial of service attacks, to lead the network work well.
4) *Logs of events records*, the actions and sensitive events in whole network, including the WEB browser, Email transmission, FTP transfers and instant message - MSN to help network administrators identify potential threats.
5) *Customize security policy*, According to the user's own network, IT professional may customize the security policy to improve the accuracy of intrusion detection.
6) *Real-time alert and response* : Multiple response modes are available in Sax2 like send console message, logs, e-mail inform, real-time cut off the connection, flexible logs.
7) *Name table:* The name table allows you to make or edit alias for addresses, ports and protocols, you may also specify the text color for a selected item. This useful feature can make packet-related information familiar and intelligible.
8) *Support multi-adapters:* If you have more than one adapter installed on the local machine, Sax2 can capture the traffic on all the adapters.
9) *In-depth packet decoding:* SAX2 provides detail packet decoding information. Conversation & Packet Stream Monitor all conversations and reconstruct packet stream [6].

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 5, May 2013

B. SAX2 architecture

It is comprised of following modules in its architecture:

- packet capturing
- matching rules
- protocol analysis
- comprehensive diagnosis
- incident response
- policy management
- logs
- display for results

The operation of Sax2 is completely dependent on analysis of internet protocols. The technology is used by Sax2 is an efficient multi-pattern matching algorithm to analyze high-speed network [7].

VII. ARCHITECTURE of SIGNATURE BASED NETWORK INTRUSION DETECTION SYSTEM

SNORT is a signature based NIDS. SNORT can be divided into five major components that are each critical to intrusion detection.

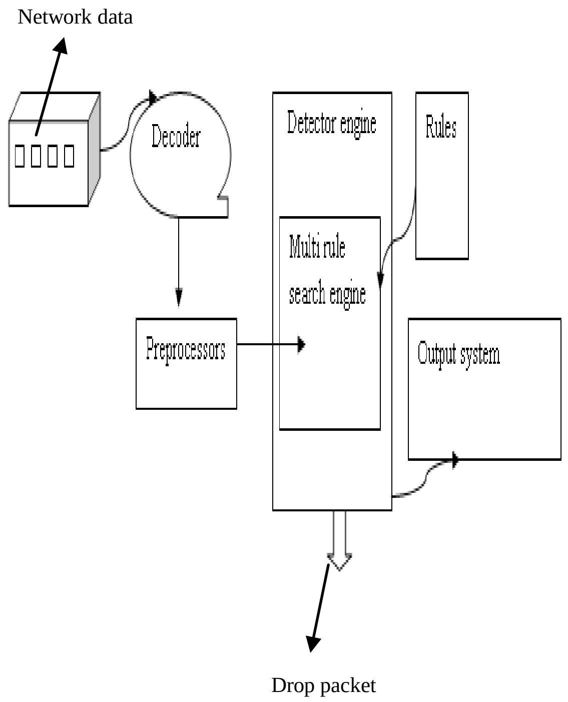

Fig2: SNORT Architecture

Fig 2 show the snort architecture. The first is the packet capturing mechanism. SNORT relies on an external packet capturing library (libpcap) to sniff packets. After packets have been captured in a raw form, they are passed into the packet decoder. The decoder is the first step into SNORT's own architecture. The packet decoder translates specific protocol elements into an internal data structure [12]. After the initial preparatory packet capture and decode is completed, traffic is handled by the preprocessors. Any numbers of pluggable preprocessors either examines or manipulate packets before handing them to the next component: the detection engine. The detection engine performs simple tests on a single aspect of each packet to detect intrusions. The last component is the output plugins, which generate alerts to present suspicious activity to you [10].

C. Comparison between NBIDS SNORT and SAX2

1) SNORT is an open source IDS and SAX2 is shareware IDPS.

2) SNORT is supporting by all the major OS. SAX2 is only supporting by Windows.

3) SNORT analysis all the protocol. SAX2 also analysis IP, TCP, UDP, HTTP, FTP, POP3, SMTP protocols etc.

4) SNORT and SAX2 are real time traffic analyzers.

5) SNORT and SAX2 are URL encoding, UDP port scan stealth port scans, packet logging and detecting signature attack.

6) SNORT throughput capability is 100mbps without packet loss. A SAX 2 throughput capability is high as compare to SNORT.

7) Rules set are flexible in SNORT so that changes are easily possible. SAX2 security rules are > 1500 but it can update rule set easily.

8) SAX2 is GUI so that the novice user can understand it. SNORT is good interface but not user friendly.

9) Attack response is very good in both systems

10) SNORT can detect the intrusion but not prevent. SAX2 is a intrusion detection and prevention system.

11) SAX 2 is GUI that can detect and prevent the intrusion. SAX2 is faster than SNORT and high throughput intrusion detection system with less packet dropping [7].

VIII. SINGLE KEYWORD PATTERN MATCHING ALGORITHM

Single keyword pattern matching algorithms are detecting the payload intrusion. String matching is finding a substring (called a pattern) within another string (called a text). Pattern and text are both strings built over a fixed and finite non empty alphabet. And give the output of all occurrences of the pattern in the text.

Keyword/ pattern is denoted as x=x[0-- - - -m-1]
m = length of the pattern.
Text/input is denoted as y=y[0- - - --- - - - -n-1]
n=length of the input[10].

A. BRUTE FORCE ALGORITHM

Brute force algorithm is a very trivial string matching algorithm. It consists in checking at each position from 0 to m-n of the text by employing a pattern of size m .This is done by comparing every character in the pattern with the corresponding character in the text. If all the characters match, then it is said to be a match or data is intruded[11].

Algorithm 1 Brute Force Single-Keyword Matching Algorithm

```
1:procedure Brute_Force(x,m,y, n)
//Input:
 //x=array of m bytes representing the keyword
 //m =integer representing the keyword length
 // y= array of n bytes representing the text input
 // n= integer representing the text length
  2:  for j = 0 to  n − m do    //every character in y
  3:  i = 0
  4:  while i < m and x[i] = y[i + j] do
  5:  i = i + 1                  // i = count of matching
  6:  end while
  7:   if i>= m then
   8:  output j
   9:  end if
  10: end for
     11:end procedure
```

Main points-Here we outline the main features of the above algorithm.

- No preprocessing phase.
- Constant space required. No extra memory required other than the memory storage for pattern and text.
- Always shifts the window by one position to the right.
- Character comparisons can be done in any order.
- Searching phase is O(mn) time complexity.

Expected character comparisons 2n[10].

Example:-

Input:AAAAAAAAHAAAAAAAAAAAAAAH
Pattern AAAAH

1)<u>AAAAA</u>AAAAHAAAAAAAAAAAAAAAH
AAAAH **5 comparisons made**
2)A<u>AAAAA</u>AAAHAAAAAAAAAAAAAAAH
AAAAH **5 comparisons made**
3) AA<u>AAAAA</u>AHAAAAAAAAAAAAAAAAH
AAAAH **5 comparisons made**
4) AAA<u>AAAAA</u>HAAAAAAAAAAAAAAAAH
AAAAH **5 comparisons made**
5) AAAA<u>AAAAH</u>AAAAAAAAAAAAAAAAH
AAAAH **5 comparisons made** Pattern is found that means input is intruded. 25 comparisons made for searching.

B. Knuth-Morris-Pratt Algorithm

Knuth have proposed a string matching algorithm that turns the search string into a finite state machine, and then runs the machine with the string to be searched as the input string. KMP is linear time algorithm for the string matching problem. A matching time of O(n) is achieved by avoiding comparisons with elements of 'S' that have previously been involved in comparison with some element of the pattern 'p' to be matched. i.e., backtracking on the string 'S' never occurs.

Components of KMP algorithm:

• The prefix function, Π

The prefix function,Π for a pattern encapsulates knowledge about how the pattern matches against shifts of itself. This information can be used to avoid useless shifts of the pattern 'p'. In other words, this enables avoiding backtracking on the string 'S'[4].

• The KMP Matcher

With string 'S', pattern 'p' and prefix function 'Π' as inputs, finds the occurrence of 'p' in 'S' and returns the number of shifts of 'p' after which occurrence is found[4].

The prefix function, Π
pseudocode computes the prefix fucnction, Π:

```
Compute-Prefix-Function (p)
m ← length[p] //'p' pattern to be matched
Π[1] ← 0
k ← 0
for q ← 2 to m
do while k > 0 and p[k+1] != p[q]
do k ← Π[k]
If p[k+1] = p[q]
then k ← k +1
Π[q] ← k
return Π [21]
```

The KMP Matcher:
The KMP Matcher, with pattern 'p', string 'S' and prefix function 'Π' as input, finds a match of p in S. pseudocode computes the matching component of KMP algorithm:

```
KMP-Matcher(S,p)
1 n ← length[S]
2 m ← length[p]
3 Π ← Compute-Prefix-Function(p)
4 q ← 0        //number of characters matched
5 for i ←1 to n        //scan S from left to right
6 do while q > 0 and p[q+1] != S[i]
7 do q ←Π[q]        //next character does not match
8 if p[q+1] = S[i]
9 then q ←q + 1        //next character matches
10 if q = m            //is all of p matched?
11 then print 'Pattern occurs with shift 1  i – m        12 q
← Π[ q]       // look for the next match[5].
```

Main points- The main points of the Knuth-Morris-Pratt algorithm are outlined below

- Performs the comparisons from left to right.
- Preprocessing phase in O(m) space and time complexity.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 5, May 2013

- Searching phase in O(n+m) time complexity.
- Matching time complexity is O(n)[11].

Example
Input AAAAAAAAAHAAAAAAAAAAAAAAH Pattern AAAAH

```
1)AAAAAAAAAHAAAAAAAAAAAAAAAH
  AAAAH 5 comparisons made
2)AAAAAAAAHAAAAAAAAAAAAAAAAH
   AAAAH  1 comparison made
3)AAAAAAAAHAAAAAAAAAAAAAAAAH
    AAAAH  1 comparison made
4)AAAAAAAAHAAAAAAAAAAAAAAAAH
     AAAAH  1 comparison made
5)AAAAAAAAHAAAAAAAAAAAAAAAAH
      AAAAH 1 comparison made
6)AAAAAAAAAHAAAAAAAAAAAAAAAH
       AAAAH          5 comparison made
```

Pattern is found after 10 comparisons which is less then brute force algorithm.

C. Comparison between BF and KMP algorithms

1) KMP Performs the comparisons from left to right and In BF Character comparisons can be done in any order.
2) KMP performs preprocessing phase in O(m) space and time complexity. In BF Searching phase is O(mn) time complexity.
3) KMP searching phase are O(n+m) and BF comparisons 2n.
4) Preprocessing phase can be done in KMP. No preprocessing phase can be done in BF.
5) BF, Constant space required. No extra memory required other than the memory storage for pattern and text. KMP need extra space and time for preprocessing [13],[14].

If the pattern is small (1 to 3 characters long) it is better to use the naive algorithm otherwise alphabet size is large the Knuth-Morris-Pratt algorithm is a good choice.

IX. LFA (LESS FALSE ALARM) ALGORITHM

False positive has become a critical factor in determining the success of IDS. False positive is an event that alarms IDS without attack being happened. So IDS development should not only focus on its capability in identifying real attacks but also on its ability to suppress the false alarm. For any system, usually false alarms generated per day outnumber the true alarms. Approximately 96% of alerts generated are asserted as false positive. I propose an algorithm, 'Less False Alarm' (LFA) that will reduce false alarm count. Intrusion can be possible on payload. So patterns are defined to detect the intrusion but it is possible that pattern is normal data. IDS system generates false alarm to identify that pattern in data. My hypothesis is that if pattern is repeated more than two times, possibility is higher that it will be intrusion instead of normal data. In such cases, the alarm will be triggered. This algorithm will create first table to reduce the no. of comparison and then compare.

Pseudo code of LFA algorithm

```
Compute_prefix_table(p)              //p for pattern
m<-length[p]                  //m=length of the pattern
T[1]<-0                        // T=table, initialize to 0
K<-0                  //simple variable for comparison
For q<-2 to m       //for loop start 2 to pattern length
Do while k>0 and p[k+1]!=p[q]              //matching
Do k<- T[k]
If p[k+1]=p[q]                                 //matching
Return T

LFA_matcher(s,p)                  //s=string, p=pattern
1.n<-length[s]                     //n=length of string
2.m<-length[p]                   //m=length of pattern
3.T<-compute_prefix_table(p)         //call function
4.q<-0                                          //variable
5.p<-0                                          //variable
6.count<-0              //variable, count pattern match
7.place[1]<-0 //array for storing the pattern position
8.for i<-1 ton        //loop for scan string from L to R
9.do while q>0 and p[q+1]!=s[i]
10.do q<-T[q]   //next char does not match check T
11.if p[q+1]=s[i]                                  //if match
12.then q<-q+1                     //shift by one character
13.if q=m          //match all the character of pattern
14.then place[p]=i-m     //pattern matching position
15.count=count+1                //increment the counter
16.end if
17.q<-T[q]                       //look  for the next match
18.end for
19.if count>=3
20.then print"intrusion";
21.print "Location",place[p];
22.else
23.print"normal data";
24.endif
25.end if
```

Time complexity of LFA algorithm is O(n). Number of comparison O(n) and in worst case O(n+m). Reduce false alarm percentage.

Example1
String: LET US ATTACK THIS PROBLEM
Pattern: ATTACK

1. **L**ETUSATTACKTHISPROBLEM
   ATTACK 1 comparison
2. L**E**TUSATTACKTHISPROBLEM
   ATTACK 1 comparison
3. LE**T**USATTACKTHISPROBLEM

ATTACK 1 comparison
4. LET**U**SATTACKTHISPROBLEM
ATTACK 1 comparison
5. LETU**S**ATTACKTHISPROBLEM
ATTACK 1 comparison
6. LETUS**ATTACK**THISPROBLEM
ATTACK 6 comparisons count+1
7. LETUSATTACK**T**HISPROBLEM
ATTACK 1 comparison
8. LETUSATTACKT**H**ISPROBLEM
ATTACK 1 comparison
9. LETUSATTACKTH**I**SPROBLEM
ATTACK 1 comparison
10.LETUSATTACKTHI**S**PROBLEM
ATTACK 1 comparison
11.LETUSATTACKTHIS**P**ROBLEM
ATTACK 1 comparison
12.LETUSATTACKTHISP**R**OBLEM
ATTACK 1comparison
Count=1
Which is <= 3 that means data is not intruded and pattern is a part of normal data .
Print"normal data".

Example 2
String: ATTACK IT ATTACK IT ATTACK IT
Pattern: ATTACK

1.ATTACKITATTACKITATTACKIT
ATTACK 6 comaprison, count+1, i-m
2.ATTACK**I**TATTACKITATTACKIT
ATTACK 1 comparison
3.ATTACKI**T**ATTACKITATTACKIT
ATTACK 1 comparison
4.ATTACKIT**ATTACK**ITATTACKIT
ATTACK 6comparison count+1,im
5.ATTACKITATTACK**I**TATTACKIT
ATTACK 1 comparison
6.ATTACKITATTACKI**T**ATTACKIT
ATTACK 1comparison
7.ATTACKITATTACKIT**ATTACK**IT
ATTACK
6 comparison, Count+1, i-m
Count=3
Which is <=3 that means input is corrupted. So alert alarm with the places where pattern found should be triggered.
Print"intrusion"
Print "location" place[p].

In example 1 pattern is the part of normal data. KMP AND BF algorithms will force to generate false alarm for intrusion. To overcome this problem, I propose LFA algorithm to reduce the false alarm count. LFA algorithm reduces false alarms without increasing time and search complexity. I analyzed that intruder will repeat the pattern for intrusion. If pattern is present one or two times, it will be the part of the data. LFA algorithm time and search complexity is similar to KMP algorithm but it gives better results. So LFA algorithm is superior in the case of reducing false alarm.

X. Conclusion

Network based Intrusion detection system can detect small attacks or stepping stone of big attack. Signature based IDS play important role in NIDS but With Time New Malicious data with New Pattern may exist, Update of the signature pattern is very important and difficult otherwise it cannot able to detect new attacks. Main challenge of IDS is to suppress false alarm. LSA algorithm will help to reduce the false alarm. It will generate alarm only when pattern is present more than two times in data otherwise ignore it.

REFERENCES

1. Sans institute infosec reading room, Understanding Intrusion Detection System, Internet, sans institute ,1 to 9, 2001.
2. Karen Scarfone, Peter Mell, Guide to Intrusion detection and prevention systems (IDPS), NIST, 1 to 127, 2007.
3. Tiwari Nitin, S. R. Singh and P. G. Singh, Intrusion Detection and Prevention System (IDPS) Technology- Network Behavior Analysis System (NBAS), International Science Congress Association , 51-56, July (2012).
4. B. Raju1 and B. SrinivasNetwork Intrusion Detection System Using KMP Pattern Matching Algorithm, IJCST,33-36, January 2012
5. C. U. Chauhan and V.A.Gulhane, Signature Based Rule Matching Technique in Network Intrusion Detection System, internet, 412-416, April 2012.
6. Faisal Mahmood , INTRUSION DEECTION SYSTEM using Sax 2.0 and wireshark 1.2.2, Internet, 1-19, 10/6/2009
7. Bhavani sunke, Research and Analysis of Network Intrusion Detection systems, Internet, 1-88, 2008.
8. David Geer, Behavior-Based Network Security Goes Mainstream, IEEE,14-17, march 2006
9. G.C.Tjhai, M.Papadaki, S.M.Furnell, N.L.Clarke, Investigating the problem of IDS false alarms An experimental study using Snort, internet, 253-267, 2008.
10. James Kelly, An Examination of Pattern Matching Algorithms for Intrusion Detection Systems, Internet,1-208, August 2006
11. SIDDHARTH SAHA, Network Intrusion Detection System Using String Matching, Internet, 1-46, 2010.
12. Martin Roesch, S n o r t — lightweight Intrusion Detection for networks, Internet, 1-11, 1999.
13. Ricardo A. Baeza-Yates, STRING SEARCHING ALGORITHMS, internet, 1-18, 1992.
14. Nimisha Singla, Deepak Garg, S tring Matching Algorithms and their Applicability in various Applications , internet, 218-222, January 2012.