Intrusion Detection in Dos Attacks

P.Rajapandian
Asst.Professor
Dept of Computer Science
Madurai Kamaraj University College
Madurai,Tamil Nadu,India

Dr.K.Alagarsamy
Associate Professor
Dept of Computer Application
Madurai Kamaraj University
Madurai,Tamil Nadu,India

ABSTRACT
Different signature or misuse based intrusion detection techniques; anomaly detection is accomplished of detecting novel attacks. However, the use of anomaly detection in practice is vulnerable by a high rate of false alarms. Pattern based techniques have been shown to make a low rate of false alarms, but are not as efficient as anomaly detection in detecting novel attacks, particularly when it comes to network probing and Denial-Of-Service (DOS) attacks. In this paper we find a new approach that merge pattern-based and anomaly-based intrusion detection, mitigating the weak point of the two approaches while increasing their strengths. Our approach begins with network protocols, and expands these state machines with information about statistics that need to be maintained to detect anomalies.

Keywords: Intrusion detection, Anomaly detection, Network monitoring, Denial-Of-Service, Misuse detection

1. INTRODUCTION
Intrusion detection moves toward can be divided into three types misuse detection, anomaly detection and pattern-based detection. Misuse detection techniques identify attacks as instances of attack signatures. This approach can detect known attacks accurately, but is ineffective against previously hidden are unknown attacks, as no signatures are available for such types of attacks.

Anomaly detection overcomes the inadequacy of misuse detection by focusing on common system behaviors, rather than attack behaviors. This approach is characterized by two steps: in the *training phase*, the behavior of the system is observed in the absence of attacks, and machine learning techniques used to create a outline of such normal behavior. In the *detection phase*, this profile is compared against the current behavior of the system, and any variations are flagged as potential attacks. Regrettably, systems often exhibit valid but previously unseen behavior, which leads anomaly detection techniques to produce a high degree of false alarms. Moreover, the efficiency of anomaly detection is affected greatly by what aspects of the system activities are learnt. The problem of selecting a suitable set of features has proved to be a hard problem. Pattern-based methods are similar to anomaly detection in that they also identify attacks as deviations from a norm. However, instead of relying on machine learning techniques, pattern based approaches are based on manually developed patterns that capture valid system behaviors.
They avoid the high rate of false alarms caused by legitimate but unseen behavior in the anomaly detection approach.

1.1 Overview of Approach
The first step in our approach is to develop provision of hosts and routers in terms of network packets received or transmitted by them. These provisions are derived from RFC's or other descriptions of protocols such as the Internet Protocol, ARP, Transmission Control Protocol and User Datagram Protocol. Consider a gateway node Internet. That connects a group local network to the as observed at the IP protocol layer. The fig 1 incorporates the following items: No IP fragmentation is reproduction, and only packets *from* the Internet (but not those items sent to the Internet) are captured. These packets may be destined for the gateway itself, in which case the state machine makes a conversion from the INIT to DONE state. Otherwise, a packet may be intended for an internal machine, in which case the gateway will first

\

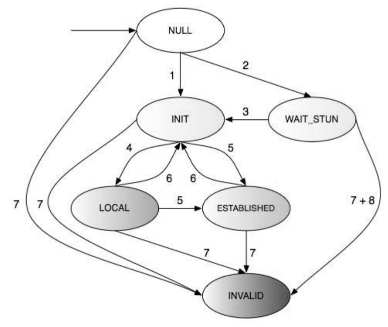

Figure 1: **pictorial representation of a pattern characterizing the gateway's behavior**

receive it on its external network interface, and make a transition from the initial INIT to packet received PKT RCVD state. Next, it will transmit the packet on its internal network interface, making a transition to the DONE state. Occasionally, the communication may not take place. This may be due to a diversity of reasons, including (a) The gateway could not determine the Media Access Control MAC address corresponding to the IP address of the target machine, (b) The gateway machine is faulty, etc. We model such condition with a timeout transition from the packet received PKT RCVD state to the DONE state. As shown in Figure 1, patterns are based on Extended Finite State Automata (EFSA). An EFSA is similar to finite state automation, with the following differences: (a) an EFSA create transitions on events that may have arguments, and (b) it can use a fixed set of state variables in which values can be stored. In the figure, we have two events, namely, pkt and timeout. The previous event denotes the reception or transmission of a packet. Their first arguments make out the network interface on which the packet was received or transmitted. The next argument captures the packet contents. The timeout denotes a time out transition, which will be taken if no other transitions are taken out of a state for a predefined period of time. The IP state machine uses two state variables src and dst. These variables are used to accumulate the source and destination IP addresses notice in a packet incoming on the gateway's external interface. By using these state variables, the state machine is able to counterpart a packet received on the external interface with the corresponding packet (it is relayed) on the internal interface.

Note that the EFSA patterns map the statistical properties of the IP packet stream to properties of traces received by the EFSA patterns. Thus, we can explain the statistical properties of the IP packet stream in terms of:
1) The rate (across traces) with which an exacting change in the EFSA is taken, e.g., the frequency with which the timeout change.
2) The sharing of values of a state variable, e.g., how repeatedly does the protocol field (in the IP header) have the value TCP, UDP, etc. In addition, we may be involved in statistical properties across a *subset* of traces, rather than all traces. The traces of interest can be specified on the origin of state variable values. For occasion, we may be interested in the number of IP-packets being relayed by the gateway to a particular local system. Based on learning statistical properties associated with the IP state system, we could detect several kinds of attacks. We describe the detection of IP sweep attack in exacting, as the detection mechanism is quite interesting. Typically, detection of IP sweep attack requires IDS to integrate knowledge about IP sweeps at some level. Often, a particular statistic is designed that specifically targets IP sweep, In contrast, we do not encode any knowledge about IP sweeps in our approach. Nonetheless, we are able to detect them as follows. Since an IP sweep attack is designed to recognize the IP addresses in use on a target network, the attacker does not know legitimate IP addresses in the target domain at the time of attack. This involve that several packets will be sent by the attacker to nonexistent hosts. This would result in a sudden burst of timeout move being taken in the IP state machine. Thus, the statistics on the occurrence of timeout transitions from the packet received PKT RCVD state can serve as a reliable indicator of the IP sweep attack.

1.2 Benefits of Approach

* P*rovides accurate attack detection. Detecting the known and unknown attacks.* Our approach can identify *all* of the attacks that were within the scope of our system. The detected attacks include very cautious attacks, e.g., port sweeps that involve 3 packets from two different hosts. As additional evidence of effectiveness, we provide beginning results in a very different context, namely, for detecting anomalies caused by email viruses. l*ittle false alarm rates.* Our system generated, on the average, 55.7 false alarms per day. This is at the low end of the false alarm, even when misuse detection based approaches (usually which we have had much lower false alarm rates compared to the anomaly detection) are taken into account.

* M*ake simpler feature selection.* One of the difficulties in anomaly detection is the choice of parameters that should be knowledge. With network packet data, there will be a large number of parameters, with many parameters assuming values from very large sets. Moreover, attack detection often requires one to believe sequences of packets. Note that the number of possible parameter arrangements across packet sequences increases rapidly with sequence length for instance, if a single packet has 10 parameters of interest, a sequence of 3 packets has a total of possible parameters. Selecting a small set of parameters from such a large space of parameters is a difficult problem. In our approach, properties of sequences are mapped into properties linked with individual transitions in the state machines. This enables us to detect most attacks by simply monitoring the distribution of frequencies with which each transition is taken.

* E*mploys redundancy to improve attack detection.* Our approach tends to learn very detailed information about many different characteristics of network protocols. Although most attacks can be detected by looking at a portion of these characteristics, the redundant characteristics benefit in two ways: An attack would likely change at least a subset of the large set of characteristics being monitored. Thus, the redundancy provides a safety against making a poor choice of characteristics to monitor, or failure in the learning algorithms. It becomes much rigid to the craft ambiguous attacks, where in an attacker attempts to carry out an attack without disturbing the parameters and features are being monitored. Clearly, it is much harder to craft attacks that preserves many different characteristics and features of the system, as opposed to just a few.

**Supports unsupervised learning.* Our techniques is robust enough to accommodate unsupervised learning, i.e., the information learnt at the end of the training phase does not need to be checked or modified by a human before it is used for detection.

2. TYPES OF INTRUSION DETECTION TECHNIQUES

Intrusion detection techniques can be broadly classified into *misuse* detection, *anomaly* detection and *pattern based* detection.

Misuse detection [20, 12], which detects known misuses accurately, is not very effective against unknown attacks. *Anomaly* detection [1, 5, 6] manages better with unknown attacks, but can generate a lot of fake positives, and hence not set up widely. Pattern-based approach [11, 23] is a recently developed technique that can detect novel attacks, while maintaining a low degree of false alarms. Most network intrusion detection systems [8, 19, 9, 13, 17, 25] reconstruct higher-level interactions between end hosts and remote users, and identify anomalous or attack behaviors. Other approaches operate mainly on the basis of packet header contents [21, 24, 22]. These techniques provide a way to define signatures not only on the basis of textual data in the reconstructed TCP sessions, but also on packet fields. These approaches can provide better detection of certain classes of attacks (especially, probing attacks) that do not result in valid TCP sessions. Our approach also relies primarily on inspecting network packer fields, but can use data in the reconstructed sessions if necessary. Data mining is concerned with the extraction of useful information from large volumes of data, thus it is natural to ask if this technique can be used to extract attack detection rules from large volumes of network traffic data. [13] was one of the first works to propose the use of data mining techniques for intrusion detection. Since then, this topic has received substantial research interest, with a lot of ongoing activity. As compared to our approach, the main difference is that these works still rely much more on expert identification of useful features for network intrusion detection. For instance, [13] selects a long list of features that include, among many others, the following: successful TCP connection, connection rejection, failure to receive SYN-ACK, spurious SYN-ACKs, duplicate ACK rate, wrong size rate, bytes sent in each direction, normal connection termination, half-closed connections, and failure to send all data packets. In our approach, we do not rely on such expert judgment to identify features, but on the protocol state machine patterns.

The Network Analysis of Anomalous Traffic Events system [24] uses statistical clustering techniques to learn normal behavior patterns in network data. Training data is used in the formation of clusters, or groups, of similar data. During detection, data points that do no t fall into some cluster are seen as anomalous. Clustering requires the use of some similarity measure and, for network data, sampling techniques are also necessary. Network Analysis of Anomalous Traffic Events system was able to detect most network probes and DOS attacks.

3. PATTERN DEVELOPMENT

Network protocols are designed through a careful and planned process. The design is captured in a exact fashion in standards documents. Such documents provide a noticeable starting point for our state machine patterns.

We have tended to theoretical from this pattern to capture only the important details of most protocols. Such information may be readily obtained from standard texts on network protocols rather than (the very longer) Internet RFCs. While strict observance to protocol standards documents is possible, this may not be desirable for two reasons. First, developing precise patterns would entail more effort than that required for more abstract patterns. Next, with strict patterns, there is always the possibility that due to smaller difference in interpretation, some traffic may be categorized as invalid by the state machine, and hence not processed properly. Furthermore, such incorrect processing may happen with some TCP implementations and not others. Using a more abstract pattern, where the state machines accept a superset of what is permitted by the values, provides a satisfactory solution to these problems. We conclude this part with a pattern of the TCP state system, as observed on a gateway connecting an organization's internal network to the internet. Our pattern is represented in Data transfer catch place in the link-ESTABLISHED state. If the TCP connection is started, from an external site, then the state machine goes through synchronization *SYN RECD* and acknowledges *ACK WAIT* states to reach the *ESTABLISHED* state. If the connection is initiated from an inside machine, then the *ESTABLISHED* state is reached through the *SYN SENT* state. In order to split down the connection, either side can send a TCP segment with the FIN bit set. If the FIN packet is transmits by an internal host, the state machine waits for an ACK of FIN to come in from the outer part. Data may keep on to be received till this ACK to the FIN is expected. It is also possible that the outer part may initiate a closing of the TCP connection. In this case we may receive a FIN, or a FIN + ACK from the external site. This scenario is represented by the states *FIN WAIT 1*, *FIN WAIT 2*, *CLOSING*, *CLOSING 1* and *CLOSING 2* states. Our state machine characterizes receive and transmit events separately, and this need additional intermediate states that are not recognized in the TCP RFCs.

If the connection termination is initiated by an external host, note that the TCP RFCs do not have the states CLOSE WAIT 1, CLOSE WAIT 2, LAST ACK 1, and LAST ACK 2 since they deal with packets observed at one of the ends of the connection. In that case, it is logical to assume that no packets will be sent by a TCP stack execution after it obtains a FIN from the other end. In our case, we are observing traffic at a middle node (gateway), so the tear down process is similar anyway of which end initiated the tear down.

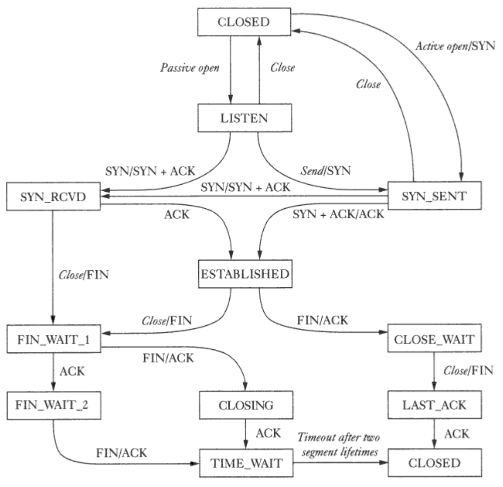

Figure 2. A new session starts the *LISTEN* state.

To decrease clutter, the following classes of abnormal transitions are not shown: (a) abnormal conditions under which a TCP connection may be terminated, including when an RST packet (with correct sequence no.) is sent by either end of the connection, as well as timeouts, (b) conditions where an irregular packet is discarded without a state transition, e.g., packets received without exact sequence numbers (after connection establishment) and packets with incorrect flag settings.

4. ANOMALY DETECTION

Information sources such as network packets fake a significant challenge for anomaly detection techniques for two reasons. First, the volume of data, and accordingly, the space of possible statistical properties of interest, is very large. Second, raw network packet data tends to be unstructured, making it complicated to distinguish meaningful information from "background noise" .To treat this problem, the raw packet data is frequently processed to extract important features that are deemed to be of interest. This process greatly reduces the amount of data to be processed by an anomaly detection system. Moreover, it classifies important information from the packet streams, while discarding less useful information. Most researchers in anomaly detection acknowledge the importance of good feature selection. At present, feature selection is driven by human expert's knowledge and judgment regarding what represents "useful information" for detecting attacks. While human experts are often in a position to find some useful features, it is far from clear that they can do a inclusive job. Often, their notion of a useful feature is influenced by their knowledge of known attacks. Accordingly, they may not necessarily select features that are useful in detection unknown attacks. In our approach, a higher degree of automation is get to the process of feature selection. Exclusively, (statistical) properties of packet sequences are mapped into (statistical) properties linked with the transitions of the state machine. Since the number of transitions is relatively small as compared to the number of possible groups of network packets, this mapping reduces the space of possible features. At the same time, our test provides evidence that this reduction does not decrease detection efficacy.

5. EXPERIMENTAL RESULTS

5.1.2 Attacks Detected by the TCP Machine

The statistics monitored by the TCP state machine is given by the following reports. Note again that we have avoided putting any great effort into feature selection. First, we have indiscriminately chosen every transition in the state machine for statistics computation. Second, we choose to this statistics collection with respect to the most obvious parameters that spot tcp sessions, namely, the source and destination addresses (or components thereof).

Table:1 Frequency in ts

1	On all frequency ts
2	On all frequency wrt (ext-ip) size 1000 ts
3	On all frequency wrt (int-ip) size 1000 ts
4	On all frequency wrt (ext-ip, int-ip) size 1000 timescale ts
5	On all frequency wrt (int-ip, int-port) size 1000 timescale ts
6	On all frequency wrt (ext-ip, int-ip, int-port) size 1000 timescale ts
7	On all frequency wrt (ext-ip, ext-port, int-ip, int-port) size 1000 t

Here ext IP and ext port refer to IP address and port information on the external network (internet), while int IP and int port refer to address and port information on the internal network. Some combinations such as (ext IP, ext port, int IP) are left out since we were most interested in traffic destined for local servers, in which case the remote port information is not useful.

6. CONCLUSIONS

In this research paper, we presented a new approach for network intrusion detection called pattern based anomaly detection. Through our experiments, we showed that the new approach combines the primary benefits of anomaly detection and pattern based detection, namely, good detection of unknown attacks and low false alarm rates. At the same, the new approach improve the principal problems associated with either approach pattern development is guided by protocol patterns, and is hence simplified. Moreover, only a handful of protocols need to be specified in order to detect most attacks. We showed that protocol patterns simplify manual feature selection process that often plays a major role in other anomaly detection approaches. In particular, most attacks discussed in the experimentation section could be detected by simply monitoring frequency distribution information associated with state machine transitions. Detection of other attacks required further partitioning of frequency information based on sources and destinations of network packets. Thus, in these experiments, our approach enables features to be selected without any significant degree of analysis or insight.

Another contribution of this paper is the pattern language for modeling state machines and for briefly stating the anomaly detection information to be learnt. This language makes it easy to apply our approach to deal with other higher layer (such as HTTP) or lower layer (e.g., Address Resolution Protocols) protocols.

7. REFERENCES

[1] Andersons. D., Lunt.T , H. Javitz, Tamaru.A, and A. Valdes, Next-Generation Intrusion Detection Expert System (NIDES): A Summary, SRI-CSL-95-07, SRI International paper, 1995.

[2] Bowen ,. T., Chee.D., Segal. M., Sekar.R., Uppuluri.P, and Shanbhag. T., Building Survivable Systems: An Integrated Approach Based on Intrusion Detection and Confinement, DISCEX 2000.

[3] Chan .P.K and Stolfo.S, Toward parallel and distributed learning by metalearning, AAAI workshop in Knowledge Discovery in Databases, 1993.

[4] Denning.D, An Intrusion Detection Model, IEEE Trans. on Software Engineering, Feb 1987.

[5] Forrest.S, Hofmeyr.S and Somayaji.A, Computer Immunology, Comm. of ACM 40(10), 1997.

[6] Ghosh.A, Schwartzbard.A and Schatz.M, Learning Program Behavior Profiles for Intrusion Detection, 1st USENIX Workshop on Intrusion Detection and Network Monitoring, 2009.

[7] Haines.J, Lippmann.R, Fried.D, Tran.E, Boswell.S and Zissman.M, 2009 DARPA Intrusion Detection System Evaluation: Design and Procedures, MIT Lincoln Laboratory Technical Report TR-1062, 2001.

[8] Heberlein.L et al, A Network Security Monitor, Symposium on Research Security and Privacy, 1990.

[9] Hochberg.J et al, NADIR: An Automated System for Detecting Network Intrusion and Misuse, Computers and Security 12(3), May 1993.

[10] Jakobson .Gand M. Weissman, Alarm Correlation, IEEE Network, Vol. 7, No. 6., 1993.

[11] Ko.C, M. Ruschitzka and K. Levitt, Execution Monitoring of Security-Critical Programs in Distributed Systems: A Pattern-Based Approach, IEEE Symposium on Security and Privacy, 1997.

[12] Kumar.S and Spafford.E, A Pattern-Matching Model for Intrusion Detection, Nat' Computer Security Conference, 1994.

[13] Lee.W and Stolfo.S, Data Mining Approaches for Intrusion Detection, USENIX Security Symposium, 1998.

[14] Lippmann.R, Fried.D, Graf.I, Haines.J, Kendall.K, McClung.D, D. Weber, S. Webster, D. Wyschogrod, R. Cunningham and M. Zissman, Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation, Proceedings of the DARPA Information Survivability Conference and Exposition, 2000.

[15] McCanne.S and Jacobson.V, The BSD Packet Filter: A New Architecture for User-level Packet Capture, Lawrence Berkeley Laboratory, Berkeley, CA, 1992.

[16] Mukherjee.B, Heberlein.L and Levitt.K Network Intrusion Detection, IEEE Network, May/June 1994.

[17] Paxson,.V Bro: A System for Detecting Network Intruders in Real-Time, USENIX Security Symposium, 1998.

[18] Porras.P and Valdes.A, Live Traffic Analysis of TCP/IP Gateways,Networks and Distributed Systems Security Symposium, 1998.

[19] Porras.P and Neumann.P, EMERALD: Event Monitoring Enabled Responses to Anomalous Live Disturbances,

National Information Systems Security Conference, 1997.

[20] Porras.P and Kemmerer.K, Penetration State Transition Analysis:A Rule based Intrusion Detection Approach, Eighth Annual Computer Security Applications Conference, 1992.

[21] Roesch.M, Snort: Lightweight intrusion detection for networks, USENIX LISA Conference, 2009.

[22] Sekar.R, Guang.Y, Shanbhag.T and Verma.S, A High-Performance Network Intrusion Detection System, ACM Computer and Communication Security Conference, 2009.

[23] Sekar.R and Uppuluri.P, Synthesizing Fast Intrusion Prevention/Detection Systems from High-Level Patterns, USENIX Security Symposium, 2009.

[24] Taylor .C and Alves.J-Foss. NATE - Networks Analysis of Anomalous Traffic Events, A Low-Cost Approach, New Security Paradigms Workshop, 2001.

[25] Vigna.G and Kemmerer.R, NetSTAT: A Network-based Intrusion Detection Approach, Computer Security Applications Conference, 1998