Fuzzy Network Profiling for Intrusion Detection

John E. Dickerson (*jedicker@iastate.edu*) and Julie A. Dickerson (*julied@iastate.edu*)

Electrical and Computer Engineering Department

Iowa State University

Ames, Iowa, 50011

Abstract

The Fuzzy Intrusion Recognition Engine (FIRE) is an anomaly-based intrusion detection system that uses fuzzy logic to assess whether malicious activity is taking place on a network. It uses simple data mining techniques to process the network input data and help expose metrics that are particularly significant to anomaly detection. These metrics are then evaluated as fuzzy sets. FIRE uses a fuzzy analysis engine to evaluate the fuzzy inputs and trigger alert levels for the security administrator. This paper describes the components in the FIRE architecture and explains their roles. Particular attention is given to explaining the benefits of data mining and how this can improve the meaningfulness of the fuzzy sets. Fuzzy rules are developed for some common intrusion detection scenarios. The results of tests with actual network data and actual malicious attacks are described. The FIRE IDS can detect a wide-range of common attack types.

1. Introduction

The Distributed Denial of Service (DDoS) attacks against major Internet sites in February 2000 highlighted the urgent need for improving the security of networks and systems connected to the Internet. In the aftermath of the DDoS attacks, security experts identified network intrusion detection as one of several technologies that can lead to improved network security [1]. While intrusion detection processes alone cannot prevent or defend against security attacks, they can serve as a valuable source of information for security administrators about the types of activity attackers may be using against them. Network intrusion detection (NID) is the process of identifying network activity that can lead to the compromise of a security policy.

Two primary forms of network intrusion detection systems (NIDS) exist: misuse detection and anomaly detection. Misuse detection relies on matching known patterns of hostile activity against databases of past attacks. Although they can be quite effective at identifying known attacks and their variants, misuse detection systems are generally unable to identify new security attacks and also require ongoing threat database updates in order to remain effective. Anomaly-based NID identifies malicious activity by applying statistical measures or artificial intelligence to compare current activity against historical knowledge of network utilization. Common problems with anomaly-based systems are that they often require extensive training data for artificial learning algorithms, and that expert systems quickly become overwhelmed with the number of rules required to identify all potential network threats.

The Fuzzy Intrusion Recognition Engine (FIRE) is an anomaly-based intrusion detection system (IDS) that uses fuzzy systems to identify malicious network activity. The system combines simple network traffic metrics with fuzzy rules to determine the likelihood of specific or general network attacks. FIRE relies on fuzzy network traffic profiles as inputs to its rule sets. Although FIRE is not exclusively a network-based detection system, we will focus on network profiling for this paper. The FIRE goals are:

- To demonstrate how fuzzy systems can be used as an intrusion detection method.
- To identify which data sources that are the best inputs to the fuzzy intrusion detection system.
- To determine the best methods for representing network input data.
- To show how the system can be scaled to distributed intrusion detection involving multiple hosts and/or networks.

Additionally, the system has been designed with two additional goals: to use readily available software and hardware as much as possible, and to be a tool accessible to the system/security administrator.

2. System Architecture

FIRE consists of the three types of components shown in Figure 1. The network data collector (NDC) is a promiscuous network data sniffer and recorder. It reads raw network packets off the wire and stores them on disk. The next component, the network data processor (NDP), summarizes and tabulates the raw packet data in carefully selected categories. In a sense, an NDP performs a kind of data mining on the collected packets. The NDP merges these summaries and tables with past data and stores them on disk. Next, the NDP compares the current data with the historical mined data to create values that reflect how the new data differs from what was observed in the past. These values are "fuzzified" to produce the fuzzy inputs needed by the Fuzzy Threat Analyzer (FTA). The resulting fuzzy inputs from the NDPs are called "fuzzy alerts" because they represent an alert condition to a degree.

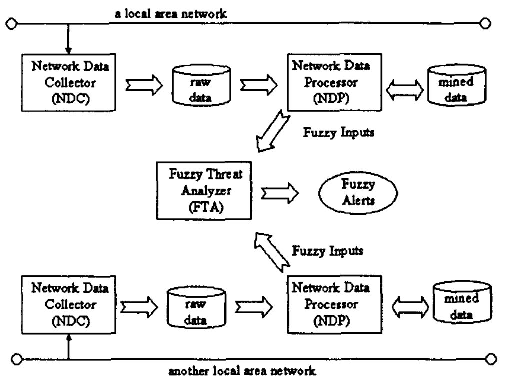
a local area network
Network Data Collector (NDC)
raw data
Network Data Processor (NDP)
mined data
Fuzzy Inputs
Fuzzy Threat Analyzer (FTA)
Fuzzy Alerts
Fuzzy Inputs
Network Data Collector (NDC)
raw data
Network Data Processor (NDP)
mined data
another local area network

Figure 1. FIRE architecture. A FIRE system includes one or more data collection units and network data processors, plus one or more Fuzzy Threat Analyzers.

The Fuzzy Threat Analyzer combines the inputs from one or more FDPs to create composite inputs. Individual NDPs can be given greater or less influence on the results by assigning them different weights. For example, an NDC located on switched network that "sees" only a few hosts might be given a smaller weighting factor than an NDC on an unswitched network that sees traffic to a greater number of hosts and, hence, has a better overall picture of the network. A rule can incorporate one or more fuzzy inputs. Depending on the fuzzy alerts they use, the fuzzy rule designer can make the types of intrusions they can detect either very general or very specific. Finally, the output from the fuzzy system executed by the FTA leads to fuzzy alerts that are sent to the security administrator for response.

The first step in any intrusion detection methodology is to identify the data feeds, or sources, of information for the IDS. In general, network data must be gathered directly from the wire. The network data collector simply grabs all packets that cross the wire and stores them to disk. No processing, filtering, or reducing of the data is performed by the NDC. After a short time interval, the NDC transfers the data gathered to the NDP for processing and then starts the next collection interval. To help avoid packet loss in the data collection system, it is important that the tasks performed by the NDC be very limited. The length of the data collection interval is chosen very carefully as a tradeoff between the need to have timely response to intrusion events and to ensure that the NDP has sufficient time to process the raw data before collecting the new data. A typical data collection interval in FIRE lasts approximately 15 minutes.

The network data processor reads the raw network data from the NDC and sorts packets by protocol into bins for TCP, UDP, and ICMP data. We will only elaborate on the data mining process for TCP here.

3. Implementation

3.1 TCP Data Mining

Data mining is the process of looking for features and relationships in data. The FIRE system applies simple data mining techniques to TCP packet data to extract metrics that are not obvious in the raw data. The metrics are chosen carefully to help reveal anomalies in the network traffic. These metrics also form the basis for the fuzzy inputs. The data mining process is essential to creating meaningful fuzzy inputs in the detection system. To gain a better understanding of the reasons for the data mining, we will examine how the TCP metrics are extracted.

We begin by extracting the following fields from each TCP header:

1. Source IP address (*src*)
2. Destination IP address (*dest*)
3. Source port number (*sport*)
4. Destination port number (*dport*)
5. TCP control bits (*tcpflag* e.g. SYN, ACK, Push, FIN, Reset)
6. Packet data length (len)
7. The date and time the packet was sent

We then apply a simple data mining technique to create an aggregate key composed of the IP source, IP destination, and destination port fields. This new field is called the *sdp*. Essentially, the *sdp* represents the existence of a TCP service channel (whether successful or not) between two IP end points. This aggregate field proves to be the most important data extracted from the TCP data. In the next data-mining phase, the Network Data Processor tabulates the number of packets seen for each *sdp* in the collection interval. The NDP also maintains a master list of unique *sdp*'s that it has seen over a pre-defined retention interval. This retention interval is usually on the order of a week or a month. Each time recently collected data is processed, data older than the retention interval is discarded. This allows the data to adjust to changes in traffic over time, while retaining some degree of history for statistical comparisons.

Not all *sdp*'s are stored. Outgoing connections to web servers outside the local network domain are ignored, simply because the quantity would be overwhelming. Additionally, only *sdp*'s with that have a destination service matching commonly used well-known TCP ports are recorded. Well-known ports are port numbers registered with the Internet Address Naming Authority (IANA). Table 1 lists the well-known TCP ports recorded by FIRE. The security administrator can modify this list as needed.

Table 1. Well-known TCP ports monitored by FIRE

Port Number	Service
21	FTP
22	SSH
23	Telnet
25	SMTP (e-mail)
53	Domain Name Service
80	HTTP
110	POP3
111	Sun RPC
113	Auth
137	NetBIOS – name service
138	NetBIOS – datagram
139	NetBIOS – session
143	IMAP
161	SNMP
177	XDMCP
194	IRC
389	LDAP
443	HTTPS
513	login
1512	WINS

The data mining process effectively reduces the size of the data that needs to be retained for future comparisons. The NDP prepares counts and other statistical measures from the mined data and stores them to disk. Since FIRE is an anomaly detection system, the measures are chosen such that anomalies in network data can be ascertained easily. Typical summaries include:

1. The number of total packets observed in the data collection interval.
2. The number of unique *sdp*'s observed in the interval.
3. The number of *sdp*'s that are new in this data collection interval.
4. The number of *sdp*'s that are new in the longer-term data retention interval (i.e. have never been seen before).
5. The number of well-known ports used in an interval.
6. The variance of the count of packets seen against the *sdp*'s.
7. The number of *sdp*'s that include foreign hosts (hosts outside the local network domain).
8. The number of successfully established TCP connections in a time interval.

The statistical variance represented by item 6 is an indicator of the distribution of hosts contacted in a time interval. Normally, one would expect to see fairly regular connections between hosts such as between a file server and its clients. Since a malicious port scan usually sends a relatively small number of packets to a large number of hosts on a network, a high variance in the packet counts across all *sdp*'s should reveal an unusual spread in the number of machines contacted in an interval.

Figure 2 shows dramatically how applying data mining techniques to the network data helps extract features that would be difficult to discern otherwise. The figure shows a 3-D plot of packet counts for each observed *sdp* during 15 minute intervals over a 3-week period from a local area network at Iowa State University. The tall, worm-like structures indicate regularly occurring service connections between two hosts on the network. The flat, horizontal structures indicate actual intruder port scans performed during this period.

The data-mining process tends to reduce the amount of data that must be retained for historical comparisons of network activity, while creating data that is more meaningful to anomaly detection than the raw input data.

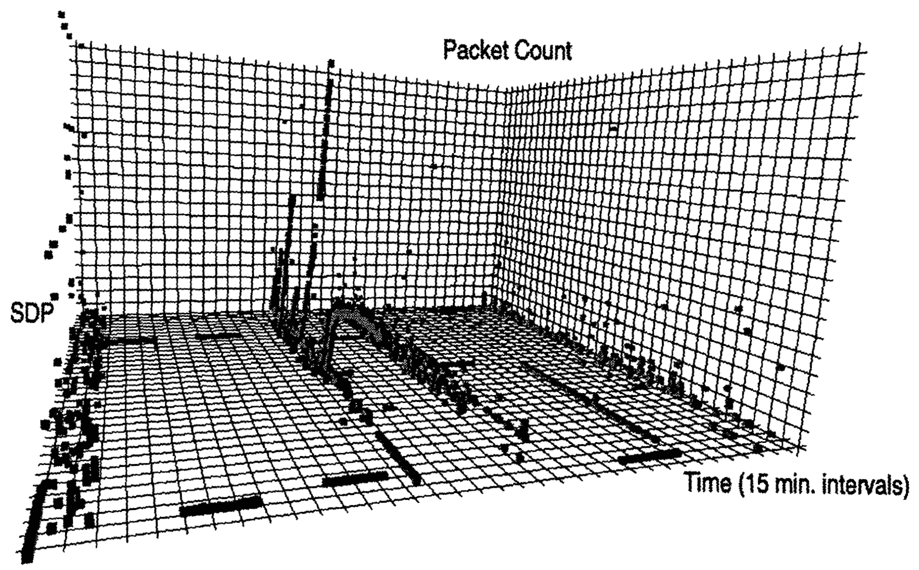

Figure 2. A three week summary of network data. The tall, worm-like structures indicate regular network connections. The flat horizontal structures indicate intruder ports scans.

3.2 Fuzzy Input Value Generation

Once the NDP completes the data mining phase, it produces fuzzy sets based on past input data. FIRE uses the historical data for each data element being monitored to calculate ranges over the input space. All of the individual data feeds are evaluated in terms of three fuzzy characteristics: COUNT, UNIQUENESS, and VARIANCE. We produce fuzzy set distributions for each of these quantities using historical data. As the data changes over time, we expect to see the input ranges of the fuzzy sets adapt accordingly.

In order to be consistent among all NDPs, it is easier if the input space for each data element can be defined with the same number of fuzzy sets. We have arbitrarily chosen five fuzzy sets for each data element: LOW, MEDIUM-LOW, MEDIUM, MEDIUM-HIGH, and HIGH. By standardizing the number of sets, we can apply the same fuzzy rules against the data from each NDP, regardless of the differences in the local input domain.

Figure 3 shows the fuzzy input sets for three types of input domains: COUNT, UNIQUENESS, and VARIANCE.

With network data, it is frequently the case that a particular data element may remain zero for a long period of time. For instance, a host on a switched network may observe no Telnet traffic for months at a time. In order to accommodate these situations, the fuzzy input sets are initially established with the LOW set covering zero in the input domain, with all other sets overlapping where the input is greater than one. This causes the MEDIUM-LOW to HIGH inputs to fire whenever any non-zero observations are made in the input data.

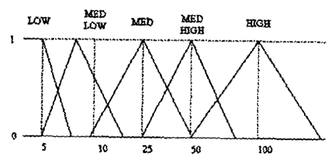

Figure 3. A typical fuzzy input set. All input domains are normalized to the same input range.

3.3 Fuzzy Rules

With the fuzzy input sets defined, the security administrator can then construct the rules of the fuzzy system. Fuzzy rules are written using common sense experiences by the security administrator. The rules designer seeks to define rules that cover as much of the input space as possible Using tools such as the Matlab Fuzzy Toolbox, the designer can check the input rule space to ensure that the fuzzy rules cover the input space and that all output responses are defined.

3.4 FIRE Scenarios

We can best understand how the fuzzy rules are used by looking at some standard attack scenarios and developing some candidate rule sets for each scenario.

Scenario 1: An Attacker Performing A Stealthy Network Scan

In a common intrusion scenario, an attacker conducts a stealthy port scan of a network, sending packets to several well-known ports (ftp, telnet, http, etc.) looking for systems that might be running those services. The presence of those services on a system gives a hint as to what vulnerabilities the attacker might try to exploit to penetrate the system. Additionally, the attacker may use scanners that can accurately identify the operating system on the target machine by examining the response of the TCP stack to carefully crafted TCP control messages. Knowledge of the services running and the host operating system is extremely valuable to the attacker because it helps to narrow the types of vulnerabilities the attacker can exploit on the systems. Port and operating system detection scanning may be a strong indicator that a more serious attack may be occurring in the future. Below are some fuzzy rules we might construct for detecting a port scan:

Fuzzy Rules for Detecting a Port Scan

Rule 1

IF (COUNT of SDPs == LOW) AND (UNIQUENESS of SDPs Observed == MEDIUM)

THEN "Port Scan" = MEDIUM-LOW

Rule 2

IF (COUNT of SDPs == MEDIUM) AND (UNIQUENESS of SDPs Observed == LOW)

THEN "Port Scan" = LOW

Rule 3

IF (COUNT of SDPs == MEDIUM) AND (UNIQUENESS of SDPs Observed == HIGH

THEN "Port Scan" == MEDIUM-LOW

Rule 4

IF (COUNT of SDPs == MEDIUM) AND (UNIQUENESS of SDPs Observed == HIGH

THEN "Port Scan" == HIGH

Rule 5

IF (COUNT of SDPs == HIGH) AND (UNIQUENESS of SDPs Observed == MEDIUM

THEN "Port Scan" == MEDIUM-HIGH

We can also develop rules for very specific intrusion scenarios, provided we have fine enough detail in the elements we are gathering. The following rule might be included in a rule set for detecting suspicious connections to a DNS server:

IF (COUNT of ForeignHosts == HIGH) AND (UNIQUENESS of DNS SDPs Observed == HIGH

THEN "DNS Scan" == HIGH

4 Results

Initial FIRE tests were performed on production local area networks in the College of Engineering at Iowa State University. NDCs were installed on both switched and unswitched network segments. Because computers on which to deploy the NDCs and NDPs were scarce, an NDC ran on the same computer as an NDP. The FIRE systems were connected to networks that were not protected from traffic outside the local network domain by using a firewall or other perimeter defense system. The initial test phase collected data for three weeks. It was not necessary to simulate intruder attacks on the network since actual intrusion attempts occurred almost daily.

Using the fuzzy rules defined in Section 3.3, FIRE was able to detect nine distinct TCP port scans and four separate ICMP (ping) scans of hosts on the network by potentially malicious attackers from outside the local network domain. Additionally, it was able to detect non-malicious port scans launched against the system from the local domain. The system also triggered HIGH alerts when seldom seen types of network traffic were observed, in agreement with the Fuzzy Rules used.

4.1 Future Work

We plan to refine our methods for generating the input ranges of the fuzzy input sets. We believe that it will be possible to detect a greater diversity of intrusions if we can derive fuzzy rules using other input domains besides network data.

We also plan to expand the scope of data feeds to add host-based anomaly detection. In particular, we plan to provide analysis of login information, e-mail, and web server activity. The data-mining techniques developed for network data should extend easily to host-based log files.

5 Conclusions

By carefully selecting specific features in the raw input data, we can create fuzzy input parameters that provide strong indications of anomalous activity on a network. Additionally, we have shown that it is possible to perform data mining in close temporal proximity to the data collection activity, without incurring a computational burden on the fuzzy input set generation process.

Fuzzy rules allow us to easily construct if-then rules that reflect common ways of describing security attacks. The types of attacks that can be described may be of a general nature or very specific, depending on the granularity of the data feeds used in the rules.

When combined with data mining of input data to reduce the size of the input data sets and to select features that highlight anomalies, fuzzy logic can be an effective means of defining network attacks.

References

1. Consensus Roadmap for Defeating Distributed Denial of Service Attacks, Global Incident Analysis Center – Special Notice. 1999-2000 SANS Institute, http://www.sans.org/ddos_roadmap.htm

2. Leo Wenke, Salvatore J. Stolfo. Data mining framework for building intrusion detection models. Proceeding of the IEEE Computer Society Symposium on Research in Security and Privacy, May 9-May 12 1999. IEEE Computer Society, p 120-132, 1999

3. Kwok-Yan Lam, Lucas Hui. Data reduction method for intrusion detection. Journal of Systems and Software, p 101-108, Apr 1996.

M. Esmali, R. Safavi-Naini, J. Pieprzyk. Computer intrusion detection and incomplete information. Journal of Science and Technology, p 49-55. Spring-Summer 1996.