Implementing an Anomaly-Based Intrusion Detection System: Focus on Internal Threat – Masquerade Attacks

John Tucker
United States Military Academy
West Point, NY 10996, USA.

Matthew K. Coughlan
United States Military Academy
West Point, NY 10996, USA.

Thomas Nelson
United States Military Academy
Department of Mathematical Sciences
Building 601 (Thayer Hall), Room 221A
Cullum Road
West Point, NY 10996, USA
706-575-2919

Benjamin Klimkowski
United States Military Academy
Department of Electrical Engineering and Computer Sciences
West Point, NY 10996, USA.

Abstract

Intrusion Detection Systems (IDS) are systems that detect actions on a network that attempt to compromise the confidentiality, integrity, or availability of a resource (Berge). In this research, we attempt to study anomaly-based IDSs. This project will attempt to determine an optimal method for detecting internal attacks. Anomaly-based IDS rely upon the ability to detect a significant difference in activity on a network during an attack from the normal activity. A key task for this research will be creating a baseline model of the normal activity of users on a network based upon our "training" data. After we achieve a baseline model, we will then test our model on data that contains possible intrusions. The internal threat portion of the paper will focus on modeling user activity in order to detect attacks, specifically masquerade attacks. Masquerade attacks are "attacks in which one system entity illegitimately poses as another entity" (Berge).

Introduction

A. Motivation

Corporations and Government Agencies continually seek new effective Intrusion Detection Systems, which are increasingly more difficult to produce. Private corporations, in addition to Government agencies, feel the need to protect their information and assets. Just recently, Facebook found one of the newest threats to their systems. Attackers, thought to be associated with the Iranian Revolution Guard, began an attack on the Facebook accounts of US State Department officials (Security Affairs). The attack consisted of both an external and internal threat. First, the attackers took control of low-level officials' Facebook accounts. They then used these accounts to launch a spear phishing campaign on higher-level State Department officials. This sophisticated attack occurred undetected for some time before Facebook realized the intrusion (Security Affairs). This attack is just one of many known attacks, and there are countless yet unknown attacks occurring every day. The security and safety of our nation relies upon the Government's ability to safeguard information. This real threat to the security of our nation highlights the importance and need for effective Intrusion Detection Systems.

II. Literature Review

A. Intrusion Detection Systems: A Background

1. Background

Intrusion Detection Systems as defined by SANS is a system which "detects actions that attempt to compromise the confidentiality, integrity, or availability of a resource." SANS reports that Intrusion Detection Systems try to "identify entities attempting to subvert in-place security controls." IDS can be divided into two groups: Network-Based IDS (NIDS) and Host-Based IDS (HIDS). Most IDS are signature-based systems. Signature-based systems use a set of rules that are derived from the target in order to identify malicious behavior. The most difficult part of creating effective signature-based systems is rule creation and management. These systems will only give alerts when a rule is broken, so they require constant updates and additions to the rule in order to be effective. Another form of IDS is anomaly-based systems. These systems are built by training the IDS on traffic data in order to create a model of normal activity. Any deviations from this model are then classified as events and are reported. This often creates a high false positive rate.

Collins (2014) discusses Intrusion Detection Systems (IDS) and some of their shortcomings. All IDS are called binary classifiers. This means that as data is processed the IDS classify the data into two categories: normal with no further action needed or the data is characteristic of an attack. There are three major obstacles that IDS face when attempting to classify data: the moral, the statistical, and the behavioral. The moral problem exists because it is extremely difficult in many situations to distinguish between normal activity and malicious attacks. The statistical problem that IDS face is that many systems are required to process incredible amounts of data and even very accurate IDS systems return a high number of false positives. The behavioral problem exists because attacks are intelligent enough that they can hide themselves from IDS. There are two methods for improving the classification effectiveness of IDS. These two methods are increasing the sensitivity and specificity. One of the ways in which this is accomplished is by reducing the amount of traffic that AN ID examines. The second method is to reduce the time an analyst needs to process an alert by fetching additional information, providing context, and identifying courses of action.

2. Insider Threat

One of the most prevalent threats that many networks face is insider threats. It is important first to understand what an insider threat is. The most common form of an insider threat is "referring to negligent insiders who accidentally harm systems or leak data due to carelessness." (Insider Threat Part 1) There are, however, a few different forms of insiders. The first of which is negligent insiders. "Negligent insiders are insiders who…have access to sensitive data and inadvertently lose control of it." (Insider Threat Part 1) Negligent insiders can be combated through multiple measures. One of the most effective is to limiting the amount of sensitive data the user can access. Encryption of data and education of users can also be effective in combating these users. (Insider Threat Part 2) The second group of insiders is the "Malicious insider." These insiders are "employees who intentionally set out to harm the organization either by stealing data or damaging systems." (Insider Threat Part 1) Most of the attacks by these insiders are either IT sabotage, fraud, or theft of intellectual property. (Insider Threat Part 1) These insiders can be limited through a system of checks and balances in which multiple people monitor the same information. Collecting user's logs and monitoring the network can be effective in deterring malicious insiders as well. (Insider Threat Part 2) The final type of insider is "Comprised Insiders." Comprised Insiders are "employees whose access credentials or computer have been compromised by an outside attacker." (Insider Threat Part 1) The only true way to prevent comprised insiders is through close monitoring of the network. (Insider Threat Part 2)

B. Anomally Based IDS

1. Machine Learning

Bhattacharyya (2014) defines machine learning as the study of methods for programming computers to learn. There are certain tasks that make machine learning valuable: 1) Problems for which there exist no human experts (detecting machine failures) 2) Problems where human experts exist, but where they are unable to explain their expertise (speech recognition) 3) Problems where phenomena are changing rapidly (stock market) 4) Applications that need to be customized for each computer user separately. Machine learning methods are divided into two categories, supervised, and unsupervised.

In supervised learning, the program needs labeled examples given by a "teacher." In unsupervised learning the program directly learns patterns from data without human intervention. Unsupervised learning assumes that normal instances are more frequent than anomalous instances and anomalous instances are statistically different from normal instances. This situation can naturally lead to large amounts of false positives from unsupervised systems.

Two supervised learning approaches are decision and regression trees and support vector machines. Decision trees are structures used to classify data. Each decision tree represents a set of rules which categorize data according to values of the attributes. A decision tree consists of nodes, leaves, and edges. A node of a tree specifies an attribute by which the data is to be partitioned. Each node has a number of edges which are labeled according to possible value of attributed the parent node. Edges connect either two nodes or a node and a leaf. Leaves are labeled with a decision value or labeled for categorization of the data. The tree starts by examining a set of cases that can be used to classify new cases. Each node of the tree contains a test. The leaf nodes then contain class labels instead of tests. Decision tree learners use a method of divide and conquer to construct a suitable tree from a training set. The decision tree partitions the data until every leaf has one case. This then allows the tree to classify all training cases.

Support Vector Machines (SVM) performs classification by constructing an N-dimensional hyper plane that optimally separates the data into different categories. SVMs are hyper plane classifiers based on linear separability. Each training vector occurs in a dot product and there is a Lagrange multiplier for each training point. The Lagrange multiplier value reflects the importance of each data point. When a maximal margin hyper plane is found, only the points that lie closest to the hyper plane are called support vectors, and only these points affect the behavior of the classifier.

The three main unsupervised mining tasks are cluster analysis, outlier mining, and mining frequent patterns, associations, and correlations. Cluster analysis groups data objects based on characteristics that describe the objects and relationships among them. This divides a set of objects into groups so that similar objects are grouped together. Outlier mining finds smaller groups of data objects that are considerably different from the rest of the data. It is used in fields such as financial fraud detection, rare gene identification, and data cleaning. Mining frequent patterns, associations, and correlations allows for the discovery of interesting associations and correlations within the data.

2. Statistical Anomaly Methods

Manikopoulos, Constantine, and Papavassiliou (2002) believe intrusion Detection Systems can be separated into two different forms: network/node (NIDS) and host (HIDS) type. The NIDA technique can be separated into two trends: misuse detection and anomaly detection. Misuse detection systems model known attacks and scan the system for occurrences of these patterns. Anomaly detection systems flag intrusions by observing significant deviations from typical or expected behavior of systems or users.

Manikopoulos, Constantine, and Papavassiliou (2002) discuss a statistical anomaly detection system they developed: Hierarchical Intrusion Detection system (HIDE). HIDE gathers data from network traffic, system logs, and hardware reports and statistically processes and analyzes the information. HIDE then detects abnormal activity patterns based on the reference models, which correspond to the expected activities of typical users, for each parameter individually, or in combined groups using neural network classification. Next, HIDE generates the system alarms and event reports. Finally, HIDE updates the system profiles based on the newly observed network patterns and system output. The general scheme of the system is that a probe collects network traffic of the host or network. It then sends this traffic to a processor, which compares the traffic with typical network activities. It then classifies the activity, as malicious or routine, and generates reports when necessary.

The paper then explores the statistical model in greater depth. Observed behavior may be described using a statistical metric and model. The period of observation may be a fixed interval of time (sec, minute, etc.), or the time between two audit-related events (i.e., between login and logout, file open and file close, etc.). Observations (sample points) of are used together with a statistical model to determine whether a new observation is abnormal. In this model, the observations are represented by a number of probability density functions. Most early IDS systems simply measured the means and variances of the monitored variables and detected whether certain thresholds were exceeded in no stationary systems that often do not follow the normal distribution, such systems generate incorrect decisions.

To overcome some of these problems, NIDES used a statistical test to determine the similarity between the expected and actual distributions. However, in real applications the data may not follow distributions; therefore, NIDES solved this problem by building an empirical probability distribution that is updated daily in real-time operation. In HIDES, however, the similarity metrics we have investigated are based on the Kolmogrov-Smirnov (K-S) test (Manikopoulos 2002). In using the K-S test, the reference models and observed system activities are represented by cumulative density functions (CDFs). What makes the K-S statistic powerful is the fact that it is distribution-independent and thus free of explicitly or implicitly assuming that the distribution is a normal one. This is a very important property for network monitoring, where very little may be known about the underlying distribution for either typical or anomalous traffic.

C. Insider Threat

1. Background on Insider Threat

Schultz (2002) explores the importance of considering internal threats, as well as external threats, when developing Intrusion Detection Methods. The first aspect of insider threats that this paper addresses is setting a definition for what an insider attack is. Insider attackers are those who are able to use a given computer system with a level of authority granted to them and who in so doing violate their organization's security policy. Others have similar definitions to this. Schultz (2002), however, believes that many definitions fail to address the difficulties in defining who is an employee, contractor, or consultant. Many attacks can be carried out by former contractors or others who may still have access to the network, and it is important to realize that these can also be considered insider attacks.

Schultz (2002) then gives brief descriptions of the previous work done in the field of insider threats. He asserts that most models claim that to commit an attack the perpetrator must have the capability to commit the attack, motive to do so, and opportunity to commit the attack. Then Schultz (2002) discuss specific models developed that attempt to address the threat that is posed when each of the necessary characteristics are met. Finally, Schultz (2002) attempts to define a model of his own. Schultz (2002) develops key indicators that can be used to identify a possible insider threat: deliberate markers, meaningful errors, preparatory behavior, correlated usage patterns, verbal behavior, and personality traits. Schultz (2002) then states that if it is possible to quantify each of the potential indicators, a mathematical equation such as a multiple regression equation that consists of a number of variables and their weightings can be formulated. This can then potentially help to predict the possibility of an insider threat.

2. Attacking an IDS

Tan (2002) attempts to describe methods for undermining Anomaly-Based IDS. Tan (2002) states that anomaly-based IDS allow you to detect two problems: attempts to exploit new or unforeseen vulnerabilities and detect abuse-of-privilege attacks. This makes anomaly-detection IDS more attractive and makes the study of their vulnerabilities very important. Tan (2002) lays out a method for undermining anomaly-based IDS called Stide. There are two main methods for undermining a system: modify the normal to look like the attack or modify the attack to make it appear as normal behavior. Normal can be determined by creating a baseline model of the network. A baseline model of a network means creating a model to describe the normal activity on a network without any attacks occurring. The first method of attack is conducted by creating a false normal behavior while developing the base line, so that intrusion behavior is incorporated into the model of normal. This allows intrusions to then be carried out undetected. Since anomaly-based IDS must undergo periodic updates to the normal model, this gives the attacker a period of time in which to incorporate an attack into the new "normal" activity on the network. The other method of attack relies upon the knowledge of the coverage of the anomaly detector in terms of anomalies. By knowing the kinds of anomalies that are or are not detectable by a given anomaly detector it is possible to modify attacks to manifest in ways that are not considered abnormal by a given anomaly detector.

Tan (2002) briefly describes how a Stide Anomaly Detector operates. It acquires a model of normal behavior by sliding a detector window of size *DW* over the training data, storing each *DW*-sized sequence in a "normal database" of sequences of size *DW*. The anomaly signal, which is the detector's response to the test data, involves a user-defined parameter known as the "locality frame" which determines the size of a temporally local region over which the number of mismatches is summed up.

The number of mismatches occurring within a locality frame is referred to as the locality frame count, and is used to determine the extent to which the test data are anomalous. Tan (2002) then describes how an attack is formulated. The first step is to analyze how the Stide detects anomalies. While analyzing this, we are able to determine which attacks the method will be able to detect. Knowing what Stide will detect we are able to formulate a foreign sequence that will not be detected by the system. The results of this analysis found that minimal foreign sequences were not detected by the Stide system. A minimal foreign sequence is foreign sequence whose proper subsequences all exist in the normal data. This provides a window of opportunity for the attacker. The paper then goes in depth into the specifics of how they implemented an attack on the Stide system.

By identifying the precise event and conditions that characterize the detection blindness for Stide and showing that real-world exploits can be modified to take advantage of such weaknesses, one is forewarned not only that such weaknesses exist, but also that they present a possible and tangible threat to the protected system. This allows us to improve the anomaly-based IDS and prevent such attacks in the future.

3. Masquerade Attack

A serious type of insider attack is a masquerade attack. A masquerade attack is defined as an attack in which one user impersonates another. It can be the most serious form of computer abuse. The typical approach for detecting a masquerader is based on the idea that masquerader activity is unusual activity that will manifest as significant excursions from normal user profiles. User profiles are constructed from monitored system-log or accounting-log data. Examples of the kinds of information derived from these logs are: time of login, physical location of login, duration of user session, cumulative CPU time, particular programs executed, and names of files accessed, user commands issued, and others. Maxion (2002) then explores some of the previous work done in this area: Bayes 1-Step Markov – This method is based on single-step transitions from one command to the next. It determines whether or not observed transition probabilities are consistent with historical probabilities (best performer, high false alarm rate). The Hybrid Multi-Step Markov toggles between a Markov model and a simple independence method depending on the portion of commands in the test data that were not observed (one of the best models). Uniqueness is based on the idea of command frequency: commands not seen in the training data may indicate a masquerade attempt (not very good, but had a low false alarm rate).

Maxion (2002) also discusses Naïve Bayes classification algorithm. It is a simple classifier known for its robustness to noise and its fast learning time. It has been very successful in text classification, in which it assigns a document to a particular class by classifying documents based upon word frequencies. Deciding whether a newspaper article is about sports, health, or politics, based on the counts of words in the article, is similar to the task of deciding whether or not a stream of commands issued at a computer terminal belongs to a particular authorized user. This model assumes that the user generates a sequence of commands, one at a time, and each with a fixed probability. The probability for each command for a given user is based on the frequency with which that command is seen in the training data. For each User X, a model of Not X can also be built using training data from all other users. The probability of the test sequence having been generated by Not X can then be assessed in the same way as the probability of its having been generated by User X. The larger the ratio of the probability of originating with X to the probability of originating with Not X, the greater the evidence in favor of assigning the test sequence to X. The exact cut-off for classification as X, that is the ratio of probabilities below which the likelihood that the sequence was generated by X is deemed too low, can be determined by a cross-validation experiment during which probability ratios for sequences which are known to have been generated by self are calculated, and the range of values these legitimate sequences cover is examined.

Masquerade attacks are a class of attacks, in which a user of a system illegitimately poses as, or assumes the identity of another legitimate user (Identity theft in financial transactions). Sommer (2011) does not focus on whether an access by some user is authorized since it assumes that the masquerader does not attempt to escalate the privileges of the stolen identity. However, Sommer (2011) assumes that the masquerader does not know the victims search behavior when using their system making it harder to mimic the user (this is supported by other research done on the topic). Sommer (2011) focuses on comparing real time actions by a user on the network with the previous actions on the network by the real user.

When dealing with masquerader attack detection, it is important to remember that the attacker already has credentials to access the system. This requires a constant monitoring of legitimate users on the network. One-class support vector machines (SVMs) are used to develop models for each user's activity.

Sommer then explores a small experiment that they designed to test this theory. Sommer had a small group download a program onto their computer and modeled this using behavior. Sommer (2011) then had three groups of people split into malicious group, benign group, and neutral group. The malicious group was told to go onto the system and collect as much data as possible. The benign group was told to work on a co-workers computer not trying to harm anything. The neutral group performed normal activity on their own computer system. The differences in the three groups were then modeled to determine if there could be a perceived difference in the activity of the malicious group and the other groups. Through the model, it was clear that the masqueraders had significantly different activity and that the model could predict a change in the way a user was acting on the network. This proved that the system was able to detect when an illegitimate user had access to the network.
Masquerade detection is often considered the most serious and challenging problem in computer security.

The typical ways in which masquerade attacks succeed include: obtaining a legitimate user's password, accessing an unattended and unlocked workstation, forging email address in messages, overtaking a computer via a network access. Masquerade detection is challenging for the following reasons: (i) masqueraders entering the system as valid users cannot be detected by the existing access control or authentication, (ii) by perfectly mimicking user's behavior, masqueraders are undetectable, and (iii) the legitimate user may be detected as a masquerader if the user's behavior changes. To enable masquerade detection, a string from a legitimate user is collected and used to generate a signature containing some attributes (features) of this user. This signature is then compared to the attributes generated from the currently monitored string of the potential masqueraders. If normal and intrusion activities are sufficiently distinct, attributes generated from the legitimate user activities will be more similar to the user's signature than those generated from the masquerader's session. Szymanski (2004) proposed a new method of recursive data mining called conceptor. In a conceptor, the first level of abstraction of input is based on repeating patterns, but the subsequent ones are based on repeating patterns of lower level abstractions. In recursive data mining, an input is first encoded into input symbols and then recursively mined for frequent patterns. It keeps building patterns of data until the pattern is complete. The process stops when no new dominant patterns are present in the transformed input. Since input is recursively mined for patterns, the pattern search can be done in each step in a limited-size sliding window, without changing the final representation of the result. Using a window speeds up the process of pattern recognition.

III. Intrusion Detection System

A. naïve bayes method background

Naïve Bayes is a method used in many different fields of research. It is a relatively simple method, while at the same time very effective. One of the more common uses of Naïve Bayes is the "Bag of Words" method (Naïve Bayes). This method attempts to classify documents into particular categories based on word frequency. The frequency of every word in the document in calculated and ordered from greatest to least. The natures of the more frequent words are then explored in order to attempt to classify the document. Exploring the most frequent words, it is easy to classify documents into different groups in a very simple way (Naïve Bayes). This method proves surprisingly effective and useful.

Using this same concept of text classification, we are attempting to classify command lines to a particular user. We look at each command of the users and the frequency with which they occur. We then look at each command and compare this to the frequency that command occurred within the data set. We are comparing the frequency to the frequency with which the command occurred in all the users on the network. This allows us to get a deeper understanding of the probability of the command originating from the specific user. If the fraction is above a set threshold, we assume that the command originated with the user. If the fraction is not above this threshold, however, we have reason to believe the command did not originate with the user and may be an intrusion. This simple comparison of frequencies of commands between a user and all users will provide a simple, effective manner for flagging intrusions.

B. Data Set

One of the most important factors of creating a truly effect Intrusion Detection System is finding a good data set. We needed a data set that modeled the true activity seen on a network during both times non-intrusion activity and intrusion activity. Due to limited time and resources, creating a new data set would have been difficult. One of the problems with using an external data set is the validity of this data set. Many data sets tend to over fit the data to the specific problem that they are exploring.

The creator of the data set will create the data in such a way to allow their model to appear accurate. This "over fitting" of the data to a method is bad because it fails to accurately depict the network. The true activity that would be seen on a network is not properly modeled and false results occur. Our model can be shaped to accurately detect intrusion on any data set, but if the data set does not model the network, it is useless.

The data set we chose to use to test our model was developed for other research. While it may be worrisome that the data set was artificially created for research, it does not prove it is a bad data set (Schonlau). When looking deeper into the data set, it becomes clear that the command lines accurately model the activity a user would have on the network. The data set was clearly not made to fit a specific method. In fact, multiple methods have been used on the same data set and proved effective. Because multiple methods were applied to the same data set, by credible researchers, we are reaffirmed in our belief that the data set is not made to over fit the data. Dr. Kwong Young, Ph.D Stanford University, as well as Ke Wang and Salvatore Stolfo at an IEEE Conference utilized this data set (Schonlau). These credible sources are a testament to the validity and usefulness of this data set. This effective data set allows us to gain an accurate gauge of the effectiveness our method will have on a true network.

The data set structure is beneficial to the development of our method. The data set consists of 15,000 commands from 50 different users. The first 5000 commands of each user do not contain any masqueraders. This will be used as our training data to develop a "baseline" for each user. The next 10,000 commands may contain masquerade data throughout. This data will serve as the test data with which we can test the effectiveness of our model in detecting masquerade attacks. Examples of segments of commands generated by users can be seen in the Figures below:

```
1  cpp
2  sh
3  xrdb
4  cpp
5  sh
6  xrdb
7  mkpts
8  env
9  csh
10 csh
11 csh
12 sh
```

Figure 1. Raw commands of a random user.

	A	B	C	D	E	F	G	H	I	J	K	L
1	1	java	.java_wr	expr	expr	dirname	basename	egrep	egrep	egrep	egrep	LEGIT
2	2	egrep	java	aacdec	cat	aiffplay	sh	aacdec	cat	aiffplay	sh	LEGIT
3	3	aacdec	cat	aiffplay	sh	netscape	netscape	netscape	netscape	netscape	netscape	LEGIT
4	4	netscape	aacdec	cat	aiffplay	sh	netscape	netscape	netscape	netscape	netscape	LEGIT

Commands broken into 10-block segments with classifications

IV. Experimental Process

A. HAM VS. SPAM

While our IDS utilizes a Naïve Bayes classification, in order to fully understand the process of implementing a Naïve Bayes classifier we explored "HAM vs SPAM" classification. HAM vs SPAM classification is the process of determining if emails are normal everyday emails or junk emails. In the book "Machine Learning for Hackers," they successfully implement a Naïve Bayes classifier that predicts HAM vs SPAM with great success. The classifier they built searches "for differences by searching through text of words that are either noticeably more likely to occur in spam messages or noticeably more likely to occur in ham messages (Conway)." The process of classifying these emails is very similar to the "Bag-of-Words" method discussed previously.

It will take all the words of the text from the emails and attempt to classify new emails based upon the history of text in both HAM and SPAM emails (Conway). The first step in the process is to extract the text from all of the known HAM and SPAM emails. This will serve as the "training data." Then the classifier builds some probabilities of text occurrence in the known classified emails. It then extracts text from emails of unknown classification. The algorithm then compares the text to the two different categories of known text probabilities and based upon similarities determines how it will classify this new email (Conway).

The book "Machine Learning for Hackers" was extremely helpful because it outlined the steps they took to building their classifier and we were able to build our own HAM vs SPAM classifier. While building the classifier it became readily apparent how similar this process could be for our IDS. The basic principles of using text, or in our case commands, and building a training model based upon known classified text to test against possible attack command text would be the same in our IDS. This process was crucial in developing a deeper understanding of Naïve Bayes that allowed us to build our classifier.

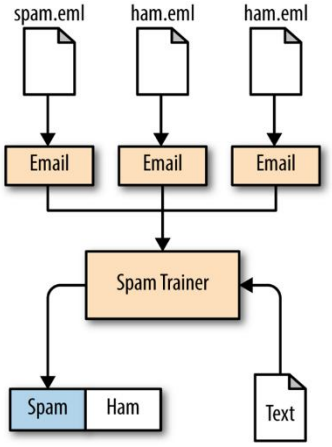

Figure 2. Methodology for classifying HAM vs SPAM

B. IDS CLASSIFIER

Once we had successfully built an effective classifier for HAM vs SPAM, we used similar concepts to build an IDS. Using the data set previously discussed, we separated the first 5,000 commands and the other 10,000 commands for each user. We then found the frequency with which each command occurred in the first 5,000 commands for each user. We also found the total frequency of each command over all the users. These frequencies would serve as the "training data" for each user and the total user commands. The next step was to calculate the probabilities of each command occurring for each user and the total. This step, however, presented a new obstacle. What would we do for commands that did not appear in the training data but appeared in the test data? It seemed reasonable that a new command could occur, and that it was necessary to have some small probability assigned to any command that had a zero frequency in the test data. We decided that any zero frequency command should be assigned a probability 10% of the lowest probability in the training data. This would allow us to continue our calculations, without running into errors dealing with zero probabilities, but also not give much weight to commands that did not occur in the training data.

The next step of building our classifier was to separate the next 10,000 commands for each user into 100 command segments. These 10,000 commands contained possible masquerade attacks and would serve as our test data. With each 100 block of commands, we would match the command with the corresponding probability of that command occurring in the training data for the user. We multiply the individual probabilities together to get the overall number for this block of commands. We would do the same process for the same 100 block of commands, but we would grab the corresponding probabilities from the training data of the total commands that we built earlier. The total commands probabilities served as the likelihood the command originated from someone other than the authorized user. We then compared our two calculations and assigned the 100 block either as a masquerade attack or as legitimate commands originating from the true user.

We replicated this process for all other 100 blocks of commands for the user. We then compared our classification to the actual classification to determine the accuracy of our classifier. We repeated this process for all 50 users. We then repeated this entire process, breaking the commands down into both 10 and 50 block of commands.

This allows us to find the optimal number of commands in each segment for our classifier. With this complete, we had completed the building and testing of a Naïve Bayes classifier that could detect masquerade attacks. The overall process of our Naïve Bayes classifier can be seen in Figure 4.

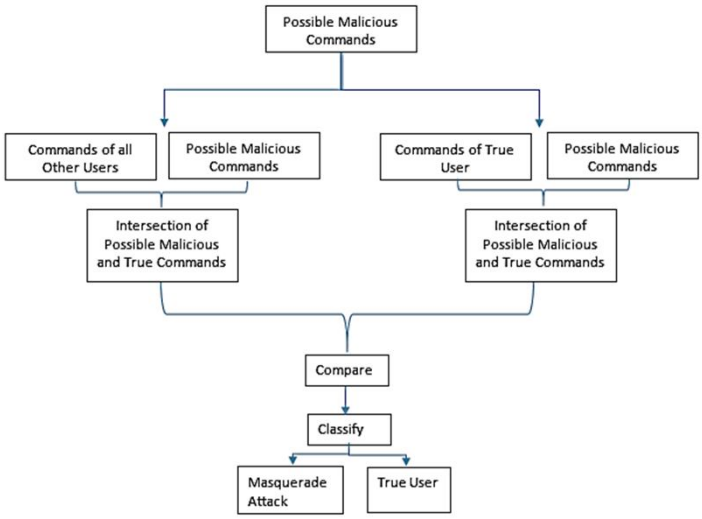

Figure 3. Methodology for implementing a Naïve Bayes IDS.

V. Results

We tested Our Naïve Bayes classifier using both "training data," consisting of commands known to have originated from the true user, and "test data," consisting of commands that could have originated from any user. The test data was broken into 100 100-command segments, 200 50-command segments, and 1,000 10-command segments, and tested against the training data of both the true user and data of all the other users. We then classified the 100, 50, and 10-command segment as either legitimate commands or a masquerade attack. After repeating this for each 100-command segment, 50-command segment, and 10-command segment for all 50 users, the average false positive and true positive rates can be seen in Table 1.

Method	**False Positive (%)**	**True Positive (%)**
10-command Segments	22.1	88.8
50-command Segments	25.4	94.2
100-command Segments	28.9	97.1

Table 1. Naïve Bayes IDS Results.

It is important to define what makes an Intrusion Detection system effective. The two measures used to measure effectiveness are false positive and true positive. False positive is classifying non-malicious traffic, or traffic originating from the true user, as malicious traffic. True positive is classifying malicious traffic, or traffic originating from an unauthorized user, correctly. These are two important measures of effectiveness of an Intrusion Detection System because they identify the system's ability to detect intrusions, but also give insight into possible over-classification which can be detrimental to the system. This gives a more robust view of our Intrusion Detection System and the effectiveness of Naïve Bayes classification.

VI. *Conclusions*

The results of our research found that our Naïve Bayes Intrusion Detection system had 88.8%, 94.2%, and 97.1% true positives. Our false positives were 22.1%, 25.4%, 28.9%. It is possible that these numbers, while seemingly high, could in fact be insignificant, and lower than necessary to classify our Intrusion Detection System as effective. In order to truly understand the significance of our results, it is useful to compare them to results of other researchers using the same data set with different classification methods:

Method	False Positive (%)	True Positive (%)
Naïve Bayes (with Updating)	1.3	61.5
Uniqueness	1.4	39.5
Bayes one-step Markov	6.7	69.3
Eigen Co-occurrence Matrix	2.5	72.3
ocSVM with Simple Commands	66.47	98.7
ocSVM with Taxonomy	60.68	94.7
10-command Segments	22.1	88.8
50-command Segments	25.4	94.2
100-command Segments	28.9	97.1

Table 2. Results from other researchers using same data set.

As seen in Table 1 above, our method resulted in very high true positive rates. This means that we were very successful in classifying true masquerade attacks as true attacks. As shown in Table 2, many of the other researches had significantly lower true positive rates, demonstrating that our results are significant. The other measure of effectiveness, false positive rate, however, is also significant. We can see Table 1 above that our false positive rates are significantly higher than many of the results shown in Table 2. This trend of high true positive rates and high false positive rates seems consistent with the only other results in Table 2 that have similar true positive rates to ours. This suggests that our method is over-predicting malicious activity. Our method appears overly cautious, and predicts suspicious activity by users as more likely malicious.

Another significant aspect of our research was determining the optimal command segment size between 10, 50, and 100-command segments. When running our Naïve Bayes classifier, we found that the manner in which our method was implemented allowed the 100-command segments to run significantly faster than the 10-command segments, and slightly faster than the 50-command segments. The significant differences in speed, along with the highest true positive, makes breaking down user commands into 100-command segments the optimal method.

In conclusion, the overall purpose for developing an Intrusion Detection System is satisfying the needs of a network administrator trying to protect his or her network. This administrator is looking for an efficient, effective Intrusion Detection System to implement on their network. Therefore, the administrator must determine whether they are willing to allocate time investigating possible non-malicious attacks, with the knowledge that they are catching almost all attacks, or if they are willing to allow some attacks to go undetected, knowing they are not wasting time investigating non-malicious attacks. An administrator must determine the correct balance of security (high true positive rates) and time available to scrutinize activity (high false positive rates). However, an administrator using the first approach, who is more concerned with a secure network, would be best to implement our Naïve Bayes Intrusion Detection System.

VII. *Future Work*

The classifier we developed presents many opportunities for future work. There were many different approaches we took which we could alter to give different results. These subtle changes allow for numerous possibilities and improvements to our model. The area of research we find very interesting for future work, however, is determining the amount of "training data" needed to create effective IDS. We used 5,000 commands, but it may happen that only 1,000 commands are necessary. An administrator on a network is trying to limit the amount of training time on a network a new user needs. This training period, in which a new user is generating a profile with each command, is a period in which an IDS cannot detect intrusions. Finding the minimum amount of training time needed to create an effective IDS allows the administrator to implement the IDS as soon as possible and limit the amount of unsecure activity on a network. Another area of future work is possibly combining our Naïve Bayes method with another method. This would create more robust IDS, which could draw upon the strengths of Naïve Bayes, high true positive rates, and the strengths of another method's low false positive rates.

List of References

"4. Naive Bayesian Classification [Book]." *Safari*. N.p., n.d. Web. 07 Apr. 2016.

Berge, Matthew. "Intrusion Detection FAQ: What Is Intrusion Detection?" *SANS*. N.p., n.d.

Bhattacharyya, Dhruba K. "An Overview of Machine Learning Methods." *Network Anomaly Detection: A Machine Learning Perspective*. N.p.: n.p., n.d. 57-120. Print

Collins, Michael. "Chapter 7: Classification and Event Tools: IDS, AV, and SEM." Network Security Through Data Analysis Building Situational Awareness. Sebastopol, CA: O'Reilly Media, 2014. 129-45. Print.

Conway, Drew, and John Myles. White. *Machine Learning for Hackers*. Sebastopol, CA: O'Reilly Media, 2012. Print.

Cross, Tom. "Insider Threats Part 1 – Who Is Attacking Your Network?" *Lancope*. N.p., 28 July 2014.

Cross, Tom. "Insider Threats Part 2 – What Can We Do about It?" *Lancope*. N.p., 04 Aug. 2014. Web.

"Facebook First Discovered Spear Phishing Attacks of Iranian Hackers on State Department Employees." *Security Affairs*. N.p., 26 Nov. 2015. Web. 03 Mar. 2016.

"Glossary of Security Terms - M." *SANS:*.N.p., n.d. Web. 03 Mar. 2016.

Manikopoulos, Constantine, and SymeonPapavassiliou. "Network intrusion and fault detection: a statistical anomaly approach." *Communications Magazine, IEEE* 40.10 (2002): 76-82.

Maxion, Roy, and Tahlia N. Townsend. "Masquerade detection using truncated command lines." *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*. IEEE, 2002.

"Matthias Schonlau's (Matt Schonlau) Homepage." *Matthias Schonlau's (Matt Schonlau) Homepage*. N.p., n.d. Web. 03 Mar. 2016.

"Naïve Bayes." *Feature Selection and Ensemble Methods for Bioinformatics Algorithmic Classification and Implementations* (n.d.): 13-31. Web.

Schultz, E. Eugene. "A framework for understanding and predicting insider attacks." *Computers & Security* 21.6 (2002): 526-531.

Sommer, Robin, DavideBalzarotti, and Gregor Maier. "Modeling User Search Behavior for Masquerade Detection." *Recent Advances in Intrusion Detection 14th International Symposium, RAID 2011, Menlo Park, CA, USA, September 20-21, 2011: Proceedings*. Heidelberg: Springer, 2011. 181-200. Print.

Szymanski, Boleslaw K., and Yongqiang Zhang. "Recursive data mining for masquerade detection and author identification." *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC*. IEEE, 2004.

Tan, Kymie MC, Kevin S. Killourhy, and Roy A. Maxion. "Undermining an anomaly-based intrusion detection system using common exploits." *Recent Advances in Intrusion Detection*. Springer Berlin Heidelberg, 2002.