Anomaly Detection Based Intrusion Detection

Dima Novikov
Department of Computer Science
Rochester Institute of Technology

Roman V. Yampolskiy
Department of Computer Science and Engineering and IGERT in GIS
University at Buffalo

Leon Reznik
Department of Computer Science
Rochester Institute of Technology

Abstract

This work is devoted to the problem of Neural Networks as means of Intrusion Detection. We show that properly trained Neural Networks are capable of fast recognition and classification of different attacks. The advantage of the taken approach allows us to demonstrate the superiority of the Neural Networks over the systems that were created by the winner of the KDD Cups competition and later researchers due to their capability to recognize an attack, to differentiate one attack from another, i.e. classify attacks, and, the most important, to detect new attacks that were not included into the training set. The results obtained through simulations indicate that it is possible to recognize attacks that the Intrusion Detection System never faced before on an acceptably high level.

1. Introduction

Most Intrusion Detection Systems (IDS) perform monitoring of a system by looking for specific "signatures" of behavior. However, using current methods, it is almost impossible to develop comprehensive-enough databases to warn of all attacks. This is for three main reasons. First, these signatures must be hand-coded. Attack signatures that are already known are coded into a database, against which the IDS checks current behavior. Such a system may be very rigid. Second, because there is a theoretically infinite number of methods and variations of attacks, an infinite size database would be required to detect all possible attacks. This, of course, is not feasible. Also, any attack that is not included in the database has the potential to cause great harm. Finally one other problem is that current methods are likely to raise many false alarms. So not only do novel attacks succeed, but legitimate use can actually be discouraged.

We investigate the benchmarks provided by the Defense Advanced Research Projects Agency (DARPA) and the International Knowledge Discovery and Data Mining Group (KDD) [1]. These benchmarks and the experience of prior researchers are utilized to create an IDS that is capable of learning attack behavior and is able to identify new attacks without system update. In other words, we create a flexible system that does not need hand-coded database of signatures, and that can define new attacks based on pattern, not fixed rules provided by a third party. Neural Networks are chosen as the means of achieving this goal. The use of Neural Networks allows us to identify an attack from the training set, also it allows us to identify new attacks, not included into the training set, and perform attack classification.

2. Intrusion Detection Overview

In the context of information systems, intrusion refers to any unauthorized access, not permitted attempt to access or damage, or malicious use of information resources. Intrusions can be categorized into two classes: anomaly intrusions and misuse intrusions [6]. Thus, intrusion detection has traditionally focused on one of two approaches: anomaly detection or misuse detection.

Anomaly detection seeks to identify activities that vary from established patterns for users, or groups of users. It typically involves the creation of knowledge bases compiled from profiles of previously monitored activities. Anomaly detection is usually achieved through one of the following:

1) Threshold detection, detecting abnormal activity on the server or network, for example abnormal consumption of the CPU for one server, or abnormal saturation of the network.

2) Statistical measures, learned from historical values.

3) Rule-based measures, with expert systems.

4) Non-linear algorithms such as Neural Networks or Genetic Algorithms [7].

The second approach, misuse detection, compares user's activities with the known behaviors of attackers attempting to penetrate a system. Anomaly detection often uses threshold monitoring to identify incidents, while misuse detection is most often accomplished using a rule-based approach. The misuse detection is usually achieved through one of the following:

1) Expert systems, containing a set of rules that describe attacks.

2) Signature verification, where attack scenarios are translated into sequences of audit events.

3) Petri nets, where known attacks are represented with graphical Petri nets.

4) Sate-transition diagrams, representing attacks with a set of goals and transitions [7].

Expert systems are the most common form of rule-based intrusion detection approaches. Unfortunately, expert systems have little or no flexibility; even minor variations in an attack sequence can affect the activity-rule comparison to a great enough degree to prevent detection. Some approaches have increased the level of abstraction of the rule-based approach in an attempt to compensate for this weakness, with a side effect of reducing the granularity of the intrusion detection process [8].

The most common method to identify intrusions is the method, which makes use of the log data generated by special software, like firewalls, or the operating system. It is possible that a manual examination of those logs would make it sufficient to detect intrusions. Analyzing the data even after an attack has taken place to decide the degree of damage sustained is trivial. This examination also plays a significant role in tracking down the intruders and recording the attack patterns for future detections. A well-designed IDS that can be used to analyze audit data for such insights makes a valuable tool for information systems.

The idea behind anomaly detection is to establish each user's normal activity profile, and to flag deviations from the established profile as possible intrusion attempts. A main issue concerning misuse detection is the signature development that includes all possible attacks to avoid false negatives, and the signature development that does not match non-intrusive activities to avoid false positives. Though, false negatives are frequently considered more serious. The selection of threshold levels is important, so that neither of the above problems is unreasonably magnified [6].

3. Experiments

3.1. Data

To conduct the experiments, it was decided to use the benchmarks of the International Knowledge Discovery and Data Mining group (KDD). These data are based on the benchmark of the Defense Advanced Research Projects Agency (DARPA) that was collected by the Lincoln Laboratory of Massachusetts Institute of Technology in 1998, and was the first initiative to provide designers of Intrusion Detection Systems with a benchmark, on which to evaluate different methodologies [1].

In order to collect these data, a simulation had been made of a factitious military network consisting of three "target" machines running various operating systems and services. Additional three machines were then used to spoof different IP addresses, thus generating traffic between different IP addresses. Finally, a sniffer was used to record all network traffic using the TCP dump format. The total simulated period was seven weeks.

Normal connections were created to profile that expected in a military network and attacks fall into one of five categories: User to Root (U2R), Remote to Local (R2L), Denial of Service (DOS), Data, and Probe. Packets information in the TCP dump files were summarized into connections. Specifically, a connection was a sequence of TCP packets starting and ending at some well defined times, between which data flows from a source IP address to a target IP address under some well defined protocol. In 1999 the original TCP dump files were preprocessed for utilization in the IDS benchmark of the International

Knowledge Discovery and Data Mining Tools Competitions [2].

The data consists of a number of basic features: Duration of the connection, Protocol type, such as TCP, UDP or ICMP, Service type, such as FTP, HTTP, Telnet, Status flag, Total bytes sent to destination host, Total bytes sent to source host, Whether source and destination addresses are the same or not, Number of wrong fragments, Number of urgent packets. Each record consists of 41 attributes and one target [4, 5]. The target value indicates the attack name. In addition to the above nine basic features, each record is also described in terms of an additional 32 derived features, falling into three categories:

1. Content features: Domain knowledge is used to assess the payload of the original TCP packets. This includes features such as the number of failed login attempts.

2. Time-based traffic features: these features are designed to capture properties that mature over a 2 second temporal window. One example of such a feature would be the number of connections to the same host over the 2 second interval.

3. Host-based traffic features: utilize a historical window estimated over the number of connections – in this case 100 – instead of time. Host based features are therefore designed to assess attacks, which span intervals longer than 2 seconds.

In order to perform formatting and optimization of the data, a tool was written that is capable of completing such operations as computing data statistics, data conversion, data optimization, neural network input creation, and other data preprocessing related assignments. Based on the results produced by the Preparation Tool, we made the following classifications: Each record consists of 41 fields and one target. The target value indicates the attack name. The data has 4,898,431 records in the dataset. 3,925,650 (80.14%) records represent attacks that fall into one of the five mentioned above categories. Total 22 attacks were identified. 972,781 (19.85%) records of normal behavior were found.

Attributes in the KDD datasets contained multiple types: integers, floats, strings, booleans, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Therefore, preprocessing took place to transform the data into the most optimal format acceptable by the neural networks.

First of all, the dataset was split into multiple files and duplicate records were removed. Each file contained records corresponding to a certain attack or normal behavior. Thus, a library of attacks was created. It was done to achieve an efficient way to format, optimize, and compose custom training and testing datasets. Second, symbolic features like attack name (23 different symbols), protocol type (three different symbols), service (70 different symbols), and flag (11 different symbols) were mapped to integer values ranging from 0 to N-1 where N is the number of symbols. Third, a certain scaling had taken place: each of the mapped features was linearly scaled to the range [0.0, 1.0]. Features having integer value ranges like duration were also scaled linearly to the range of [0, 1]. All other features were either Boolean, like logged_in, having values (0 or 1), or continuous, like diff_srv_rate, in the range of [0, 1]. No scaling was necessary for these attributes.

Attacks with the most number of records were chosen to be in the training set. The following attacks were used to train and to test the neural networks: Smuf, Satan, Neptune, Ipsweep, Back. The following attacks were chosen for the unknown (not trained) set of attacks: Buffer_overflow, Guess_password, Nmap, Teardrop, Warezclient.

3.2. Neural Networks Based Intrusion Detection System Experiments

It was decided to run the experiments in three stages. In stage one, it was important to repeat the experiments of other researchers and have the Neural Networks to identify an attack. In stage two the experiment was aimed at a more complicated goal. It was decided to classify the attacks, thus, the Neural Networks had to determine not only the presence of an attack, but the attack itself. Stage three had to repeat the experiments of stage two, but in this stage a set of unknown attacks are added to the testing set. Stage three contains experiments of a higher complexity and interest.

Each Radial Bases Function (RBF) Neural Network had 41 inputs, corresponding to each attribute in the dataset, two outputs for attack detection (the first output for the presence of an attack

– “YES”, the second output for the normal behavior – “NO”), or six outputs for attack classification (five outputs for the attacks, and the sixth output for the normal behavior), three layers (input, hidden, and output). The training set consisted of 4000 records. The attack and the normal behavior records were evenly distributed in the training set.

The parameters of the Multiple Layer Perceptron (MLP) NN were: 41 inputs, corresponding to each attribute in the dataset. Two outputs for attack detection (the first output for the presence of an attack – “YES”, the second output for the normal behavior – “NO”), or six outputs for attack classification (five outputs for the attacks, and the sixth output for the normal behavior). Three layers (input, hidden, and output). The hidden layer has 20 nodes, alpha = 0.7, beta = 0.8, “tansig” function is used in the input layer node, “purelin” in the hidden and output layer nodes, 50 epochs. The training set consisted of 4000 records. The attack and the normal behavior records were evenly distributed in the training set.

3.3. Results

The first stage of the experiments consisted of 2 phases. First, only one attack was used in the training set. The distribution of an attack and normal records was 50% - 50%. Table 1 represents the results of these experiments. As it is shown, the accuracy of positive recognition is very high for both Neural Networks. All of the attacks have more than 90% of recognition. Most of them are very close to 100%, what is a very good and expected result.

Table 1. One Attack Dataset Results

Attack Name	RBF Accuracy	RBF False Alarms	MLP Accuracy	MLP False Alarms
Smurf	100%	0	99.5%	0
Neptune	100%	0	100%	0
Satan	91%	7%	97.2%	2%
IP Sweep	99.5%	0	99.9%	0
Back	100%	0	100%	0

For the second phase of the first stage of the experiments, five different attacks were used in the training set. Normal behavior records was considered as an attack, thus total of six attacks were used in this stage. In order to proceed to the next level of the experiments, attack classification, it was important to prove that the attacks are distinguishable. Therefore, six different experiments were held to prove this idea. 50% of the training set consisted of the concentrated attack, i.e. the attack that had to be differentiated from the others.

Other 50% were evenly distributed between other attacks, i.e. 10% per attack. For example, normal behavior records needed to be defined. 50% of the training set for this assignment consisted of the records of normal behavior and other 50% contained records of Smurf, Neptune, Satan, IP Sweep, and Back attacks. All records were in random order.

Table 2 demonstrates the results of this experiment. As shown in the table, the accuracy for differentiating the attacks is quite high for both Neural Networks. The lowest accuracy is 91% for Satan and the highest is 100% for Smurf, Neptune, and Back. These results let us make a conclusion that attacks can be differentiated, thus classified.

Table 2. Five Attack Dataset Results.

Attack Name	RBF Accuracy	RBF False Alarms	MLP Accuracy	MLP False Alarms
Smurf	100%	0	99.5%	0
Neptune	100%	0	100%	0
Satan	91%	7%	97.2%	2%
IP Sweep	99.5%	0	99.9%	0
Back	100%	0	100%	0
Normal	98.0%	1%	96.8%	2%

For the second stage of the experiments Neural Networks with six outputs were used. At this level there was an attempt to create an Intrusion Detection System that is capable of classifying the attacks. A dataset of five attacks and normal behavior records were used. The attacks were evenly distributed in the dataset. Table 3 demonstrates the result of this experiment. As we can see the accuracy of classifying attacks is 93.2% using RBF Neural Network and 92.2% using MLP Neural Network.

The results were very close and the difference is statistically insignificant. In most cases the Networks managed to classify an attack correctly. The false alarm rate (false positive) is very low in both cases, missed attacks rate (false negative) is not high either, and the misidentified attacks rate (misclassification of the attacks) is 5%-6%. Overall, it is possible to conclude that both Neural Networks managed to accomplish the second stage of the experiments and were capable of classifying the attacks. Therefore, the environment for the third stage of the experiments was set.

Table 3. Attacks Classification

	Accuracy	False Alarms	Missed Attacks	Mis-identified Attacks
RBF	93.2%	0.8%	0.6%	5.4%
MLP	92.2%	0	2.1%	5.7%

For the final stage of the experiments we used the trained NN from the second stage. The Networks were trained to classify the following attacks: Smurf, Neptune, Satan, IP Sweep, Back, and Normal behavior records. At this point we proceeded with the most interesting and exciting phase of the experiments – untrained (unknown) attack identification.

As it was mentioned earlier, five attacks were chosen to be used for this purpose: Buffer Overflow, Guess Password, NMap, Teardrop, and Warezclient. Datasets of these attacks were sent into the trained Neural Networks. Table 4 demonstrates the results: RBF neural network managed to identify the unknown attacks as one of the trained attacks in most cases. As for the MLP Neural Network, it succeeded only with NMap and Guess Password attacks. In other cases it identified the attacks as normal behavior. Thus, RBF displayed more capabilities in identifying unknown attacks while MLP failed in some cases.

Table 4. Unknown Attacks Identification.

Attack Name	MLP	RBF
Buffer Overflow	53.3%	96.6%
Guess Password	96.2%	100%
NMap	99.5%	100%
Teardrop	1%	84.9%
Warezclient	8%	94.3%

As the previous research indicates, there were many attempts to detect and classify attacks. The winner of the last KDD intrusion detection competition, Dr. Bernhard Pfahringer of the Austrian Research Institute for Artificial Intelligence, used C5 decision trees, the second-place performance was achieved by Itzhak Levin from LLSoft using Kernel Miner tool, and the third-place contestants, V. Miheev, A. Vopilov, and I. Shabalin of the company MP13, used a decision tree based expert system [3]. Also, we note the results of the most recent research made by Maheshkumar Sabhnani and Gursel Serpen of the Ohio University who used a multi classify model to achieve even better results than the winner of the KDD Cups contest [9].

Table 5 compares the mentioned above results. As we can see, in some cases accuracy of the classification is as low as 8.4%, which is totally not acceptable. The main problem with the approach they had chosen was that they used all attacks in the dataset, though many of those attacks did not have enough records for training, as we outlined after the data formatting and optimization took place. If an attack does not have enough presence (IMAP attack had only 12 records), it should not be used for training.

Also, they grouped the attacks, what potentially can lead to a misdetection since not all of the attacks in the same group have identical signatures and patterns. Thus, a different approach was chosen to detect and classify attack. The main advantage of this approach was data formatting and the training dataset grouping, which allowed us to increase the accuracy rate up to 100% in some cases, and to achieve a high percentage of identification of the attacks that were not included into the training set.

Table 5. Result Comparison.

		Probe	DoS	U2R	R2L
KDD Cup Winner	*Accuracy*	83.3%	97.1%	13.2%	8.4%
	False Alarms	0.6%	0.3%	0.1%	0.1%
KDD Cup Runner Up	*Accuracy*	83.3%	97.1%	13.2%	8.4%
	False Alarms	0.6%	0.3%	0.1%	0.1%
Multi-Classifier	*Accuracy*	88.7%	97.3%	29.8%	9.6%
	False Alarms	0.4%	0.4%	0.4%	0.1%

4. Conclusions

Modern commercially used Intrusion Detection Systems employ the techniques of expert systems that require constant updates from the vendors. These updates make the IDS static, not flexible, and not capable of detecting new attacks without new batches. To improve the security, a lot of researchers put efforts to utilize Artificial Intelligence techniques in the area of Intrusion Detection, in order to create systems capable of detecting unknown attacks, or/and learning new patterns by themselves.

Benchmarks were created to standardize and compare the work of different investigators of this problem. Competitions were held to attract the attention of new researchers. In the most cases Neural Networks were used to detect attacks, and decision-making trees were used to classify them. After extensive study, we decided to come up with a unique solution, and approached the problem with a new dataset formatting and optimization technique.

A library of attacks was created. This library was based on the benchmark provided by the MIT Lincoln Lab that was optimized by the KDD Cups. After the data was carefully formatted and optimized, it was decided to use and compare two different Neural Networks in attack detection and classification. Neural Networks were chosen due to their abilities to learn and classify. Trained Neural Networks can make decisions quickly, making it possible to use them in real-time detection.

Both types of Neural Networks managed to perform well on the known set of attacks, i.e. attacks that they were trained to identify and classify. After new attacks were added to the testing set, i.e. attacks that were not included into the training set, Radial Basis Function Neural Network performed significantly better than Multiple Layer Perceptron with the detection rate between 80% and 100%, and the false alarm rate not greater than 2%.

When we compared these results to the results of previous work, it was notable that the chosen technique had its advantages. First of all, we managed to correctly detect the attacks. Second, classification of the trained attacks was successful with the rate of 90-100%. Third, and the most important, we were able to detect new unknown attacks, which were not included into the training set. The accuracy of detecting new unknown attacks was between 80% and 100%.

After performing our experiments we concluded that with appropriate data formatting, optimization, and dataset composition, Neural Networks display a very good performance and potential in detecting and classifying trained attack, as well as new unknown attacks that were not included into the training set. Thus, the main goal of this research was accomplished.

In the future we would like to investigate possibility of utilizing other types of neural networks to the task of intrusion detection. Additionally we would like to attempt to classify not just detect previously unknown problems, perhaps with a self-organizing neural network.

5. Acknowledgements

We would like to thank Dr. Roger Gaborski and Dr. Hans-Peter Bischof from Rochester Institute of Technology for valuables comments on our research methodology. This paper is partially based upon work supported by National Science Foundation Grant No. DGE 0333417 "Integrative Geographic Information Science Traineeship Program", awarded to the University at Buffalo.

6. References

[1] DARPA, *Intrusion Detection Evaluation.* MIT Lincoln Laboratory, 1998 (http://www.ll.mit.edu/ist/ideval).

[2] Hettich, S. and S.D. Bay, *The UCI KDD Archive.* University of California, Department of Information and Computer Science, 1999.

[3] KDD, *http://wwwcse.ucsd.edu/users/elkan /clresults.html.* KDD Cups 99 - Intrusion Detection Contest, 1999.

[4] Lee, W., S. Stolfo, and K. Mok, *Mining in a Data-Flow Environment: Eperience in Network Intrusion Detection.* In Proceedings of the 5th ACM SIGKDD, 1999.

[5] Lee, W., S.J. Stolfo, and K.W. Mok, *A Data Mining Framework for Building Intrusion Detection Models.* IEEE Symposium on Security and Privacy, 1999.

[6] Mukkamala, S., G. Janoski, and A. Sung, *Intrusion Detection Using Neural Networks and Support Vector Machine.* IEEE, 2002.

[7] Planquart, J., *Application of Neural Networks to Intrusion Detection.* SANS Institute, 2001.

[8] Rhodes, B., J. Mahaffey, and J. Cannady, *Multiple Self-Organizing Maps for Intrusion Detection.* GIT Information Technology and Telecommunications Laboratorys, 1999.

[9] Sabhnani, M. and G. Serpen, *Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context.* EECS, University of Toledo, 2003.