B-DIDS: MINING ANOMALIES IN A BIG-DISTRIBUTED INTRUSION DETECTION SYSTEM

Vandana P. Janeja, Ali Azari,
Josephine M. Namayanja
University of Maryland, Baltimore County
e-mail: {vjaneja, azari2, jona1}@umbc.edu

Brian Heilig
HAMR Analytic Technologies, E.T. International
e-mail: bheilig@etinternational.com

Abstract—The focus of this paper is to present the architecture of a Big-distributed Intrusion Detection System (B-dIDS) to discover multi-pronged attacks which are anomalies existing across multiple subnets in a distributed network. The B-dIDS is composed of two key components: a big data processing engine and an analytics engine. The big data processing is performed through HAMR, which is a next generation in-memory MapReduce engine. HAMR has reported high speedups over existing big data solutions across several analytics algorithms. The analytics engine comprises a novel ensemble algorithm, which extracts training data from clusters of the multiple IDS alarms. The clustering is utilized as a preprocessing step to re-label the datasets based on their high similarity to known potential attacks. The overall aim is to predict multi-pronged attacks that are spread across multiple subnets but can be missed if not evaluated in an integrated manner.

Keywords—big data; distributed Intrusion Detection System; Ensemble learning;

INTRODUCTION

Intrusion detection aims to identify unauthorized access using known attack signatures. The focus of this work is the discovery of multi-pronged attacks, which may be spread out over time and several points in the network. Multi-pronged attack detection is a challenging task particularly because the datasets become massive.

Example Scenario: Consider the flow of such a multi-pronged attack, which occurred at the Pacific Northwest National Laboratory in July 2011[6]. Despite the lab’s well protected IT security perimeter, the attacks made it through in a very coordinated and prolonged process. First there was an attack on the organization and second there was an attack on a partner that shares key resources. In the first part of the attack the intruders took advantage of vulnerabilities in public-facing web servers. In addition hackers secretly scouted the network from compromised workstations that had been targeted beforehand as part of a coordinated prolonged attack. The second part of the attack started with spear-phishing where a second group of hackers instituted a phishing attack on the organization’s major business partners with which it shares network resources. The hackers were able to obtain a privileged account and compromise a root domain controller that is shared by the organization and its partner. When the intruders tried to recreate and assign privileges the alarm was finally triggered alerting the organization’s cybersecurity team.

As the example illustrates, simply looking at one dimension of the data is not enough in such prolonged and multi-source attack scenarios. This is also evident in advanced persistent threat detection [1]. We leverage the concept of a Distributed Intrusion Detection System (dIDS), which provides the infrastructure for the detection of a coordinated attack against an organization and its partners’ distributed network resources. However, given the complexity of multi-source attack sources and the massive amount of data generated for such a multi-pronged attack, we propose a multi-level mining framework in the form of a big distributed IDS (B-dIDS). One important piece of data collected in most organizations is IDS log data such as Snort logs. We use the IDS logs to sift through alarms that may look benign individually but may indicate a critical alert in combination with other alerts [2].

In our proposed distributed environment each subnet contains a dIDS agent that performs local intrusion detection and generates IDS log data. Log data from each dIDS agent is sent to a control center where it is stored for aggregated analysis. Each signature-based agent generates a priority level associated with the alarm when an attack against a known threat is detected, and generates high, medium and low priority alarms for otherwise ‘anomalous’ behavior. High priority alarms can be clearly labelled, however the low and medium priority alarm data is very large making it difficult to perform manual analysis by an administrator. This data can be much larger in a large network with high traffic throughput. System administrators view the alarm data through queries to detect suspicious behavior within the network. However, in such a review several alarms that are part of a coordinated attack will be missed.

We consider this as a class imbalanced learning problem. We use an ensemble classification technique to automatically classify the large volume of aggregated alarm data and to alert a system administrator to a potential coordinated attack. We consider each agent to provide a training set that is generated after preprocessing the data through a clustering algorithm [3]. As a result the training tuples are selected from each cluster using a Split Ratio (SR) such that training sets are made up of tuples near, far or outliers with respect to the centroid of the cluster. Such a targeted selection has produced highly accurate results in scenarios where there is high sample diversity as in the case of intrusion detection where the intrusions are a minority class. In this research we explore advances in machine
learning techniques using advanced big data processing tools.

For the processing of the massive datasets within and across the distributed IDS sensors we use HAMR, which is a next generation in-memory MapReduce engine developed by HAMR Analytic Technologies (HAMRTech) with seamless support for batch and streaming analytics [4, 5]. HAMR supports the MapReduce programming model while executing machine learning algorithms and produces high speed ups over Hadoop.

**APPROACH**

We briefly outline our proposed approach for mining the multiple IDS alarms, which are submitted from multiple subnets to predict a multi-pronged attack, as shown in figure 1. Our architecture consists of the big data analytics processing through HAMR[4] and the analytics engine which will reside on HAMR:

![Figure 1: B-dIDS Control Center Analytics](image)

**A. Analytics Engine**

**1. Clustering and Labelling IDS Alarms**

We perform clustering of the IDS log data after preprocessing. The data is in the form of parsed IDS alarms where each alarm is a data point with a priority level: high, medium or low. This data is collected from multiple IDS sources as shown in figure 1. Our premise is that a low priority alarm may indicate a coordinated attack when seen in conjunction with some of the other alarms from different IDSs. Since we are looking at all alarms from multiple IDSs we have the opportunity to study the similarity between alarms and then judge whether an alarm is truly a low priority or could be potentially high priority. The end goal is to collapse and reclassify correlated alarms across multiple IDSs that could potentially indicate cyber-attacks.

All the alarms are gathered in the control subnet and clustering is performed on them. It should be noted that the traditional clustering can be performed on the numeric attributes and categorical attributes separately. The idea is that if a low priority alarm ‘l’ falls into the same cluster $C^h$ as multiple high priority alarms ‘h’ then there is a high likelihood that an IDS mislabeled the alarm ‘l’. This can further be validated by (a) small distance of ‘l’ from the cluster centroid of the high priority cluster $C^h$ which will indicate that ‘l’ is indeed close to the centroid and thus highly similar to other ‘h’ alarms in $C^h$, (b) The overall Sum of Squared Error (SSE) of the cluster $C^h$ which will indicate the quality of the clustering. High SSE indicates that the data points are spread out widely around the centroid whereas a low SSE indicates that the points are tightly knit around the centroid. Lower SSE will provide more validity to our claim that ‘l’ was indeed mislabeled. In addition, associations can be performed to identify data points with high levels of associativity to increase the likelihood of detecting a misclassified alarm, particularly if a low priority alarm consistently associates with a high priority alarm.

If these multiple or majority validations are positive that ‘l’ is mislabeled then we provide a new label to this alarm as ‘p’ which indicates that an alarm is positively indicative of an attack. Other alarms, which do not pass this test are labelled as ‘n’ meaning that they are negative data points. We use these new labels as feeding into our classification ensemble such that any new incoming alarms are predicted to be ‘p’ or ‘n’ based on this meta knowledge that we derive from multiple IDSs.

**2. Training Set Generation**

In order to create the training sets we will use the knowledge of the distribution of points in the datasets. To do this we use our clusters and extract points close to centroid, points far from centroid, and points, which appear to be outliers. This will give us a good representation of the data points (alarms) which are highly similar to the cluster centroid and those which are outliers. Data points, which are highly similar emulate the average behavior of the cluster, while points which are outliers have extreme behavior as compared to the rest of the data points.

**3. Ensemble Learning**

Our proposed ensemble classifier maximizes the diversity by creating selective subsets of instances that are similar to each other but dissimilar from those in other subsets. Each subset is created from the instances that are already similar enough to be put into the same cluster by k-means clustering. Literature suggests that the ensemble diversity has a positive influence on imbalanced class performance [12]. Therefore, we aim to attain high diversity through multiple non-overlapping training sets in the ensemble.

We create two training sets named Near and Far. These are defined based on a split ratio that shows the percentage of the examples from each cluster that should go to Near/Far.
training sets. For example, a Split Ratio (SR) value of 40% shows that 40% of the examples nearest to the cluster center go into near training set and the remaining 60% go into the Far training set. Next each training set is used to train a weak classifier (C4.5 in this case). Each trained classifier is then used to classify the same non-overlapping test set. The overall performance measures are calculated for each classifier separately. Finally, we combine the predictions made by the classifiers using a voting system; we give a weight to each classifier within the ensemble based on their overall performance for each class, and then output the predicted class label.

B. Big Data Processing

Both the model development and the classifier require a big data solution, but for different reasons. In the model development phase the static dataset will consist of IDS alarm data collected from many IDS agents over a period of several years, and can be in the 10 TB range. The volume of data combined with the complex algorithms necessary to perform the model training necessitates a distributed in-memory solution in order for the model development to complete in a reasonable amount of time. However, in the real-time system distributed IDS agents will be sending alarms to a central analysis server. The velocity of alarm data will be integrated into the B-dIDS to classify the aggregated alarm data, providing summarized alarms to system administrators of potential coordinated attacks against network resources with high accuracy.

CONCLUSIONS

In this paper, we have proposed a Big-distributed Intrusion Detection System. In this architecture, we have utilized HAMR, a big data processing engine, and proposed a novel ensemble method to identify multi-pronged attacks. We plan to experiment extensively in public datasets and provide a benchmark against existing big data solutions in terms of speedups and the quality of results.

References

[3] A. Azari, V.P. Janeja, Alex Mohseni, Healthcare Data Mining: Predicting Hospital Length of Stay (PHLOS), Accepted to the International Journal of Knowledge Discovery in Bioinformatics (IJKDB), Nov 2012
[7] HAMRTech Benchmark Report. 2, Oct 2014 Available upon request from hamr-info@hamrtech.com