Teja Kommineni
teja.kommineni@utah.edu

Robert Ricci
ricci@cs.utah.edu

School of Computing
University of Utah
50 S. Central Campus Drive
Salt Lake City, UT - 84112

ABSTRACT

Conventionally, network applications are identified using well-known ports as defined by IANA. These methods are no longer useful to identify the traffic accurately as the masqueraders invent new techniques to obfuscate their network traffic from detection. This has led to study of new ways to investigate the traffic like packet inspection and flow inspection. Though, packet inspection gives accurate results it is daunted by the heavy costs of looking into each packet of network and is unpractical in modern-day networks. This is where flow inspection comes to rescue. In this paper, we inspect the Netflow records collected on the routers to aggregate information based on host and apply ML/DM procedures to determine the behavior of these hosts rather than classifying them based on the application. This approach of extracting behaviors from network data helped us in gaining interesting insights into users of the system which we later used to build recommendation systems.

1. INTRODUCTION

This paper focuses on data mining in Networks. We apply different data mining techniques on the monitoring data recorded in computer networks.

The remainder of paper is organized as follows: Section 2 covers background, including related work in this area. Section 3 provides in detail explanation of our implementation.

2. BACKGROUND

Machine Learning and Data Mining have become prevalent in the area of networking to gather insights from the collected network data.

Machine Learning is a method of generating rules from past data to predict the future. A Machine Learning approach usually consists of two phases: training and testing. The following steps are usually performed:

- Identifying attributes (features) and classes from training data.
- Choosing a subset of the attributes for classification.
- Choosing a ML algorithm to create a model using the training data.
- Use the trained model to classify the unknown data.

Data Mining is the process of extracting implicit, previously unknown and potentially useful information from data. Data mining is generally considered as a step in the process of Knowledge Discovery from Data (KDD).

KDD usually consists of the following steps:

- Selection of raw data from which knowledge has to be extracted.
- Preprocessing the data to perform cleaning and filtering to avoid noise.
- Transforming the data so that all the attributes in the raw data are aligned towards achieving the common goal.
- Applying Data Mining algorithms to find rules or patterns.
- Interpret the observations of the above step and validate them.

There is a significant overlap between these two techniques. ML/DM approaches come in 3 forms: unsupervised, semi-supervised and supervised. In unsupervised learning problems, the main task is to find patterns, structures or knowledge from unlabeled data. If a part of data is labeled then the problem is called semi-supervised learning. If all the data is completely labeled then it is called supervised learning.

2.0.1 Netflow

NetFlow is a proprietary protocol for collecting IP flow packets information on networks. Each record in a NetFlow is called a flow. Each packet that is exchanged within a router or switch is examined for a set of IP attributes. These attributes serve as the identity of that packet.All packets with the same source/destination IP address, source/destination ports, protocol interface and class of service are grouped into a flow and then packets and bytes are tallied. This form of aggregating packets to flow has an advantage because it decreases the amount of information needed to be stored in the database.

The captured flows are exported to NetFlow collectors at frequent intervals. The flows that are to be exported are

NetFlow Data – Flow statistics	
Source IP address	IP address of the device that transmitted the packet.
Destination IP address	IP address of the device that received the packet.
Source Port	Port used on the transmitting side to send this packet.
Destination Port	Port that received this packet on the destination device.
TCP Flags	Result of bitwise OR of TCP flags from all packets in the flow.
Bytes	Number of bytes associated with an IP Flow
Packets	Number of packets associated with an IP Flow

Figure 1: NetFlow Fields captured in a Flow

determined based on the following rules: 1) when it is inactive for a certain time without receiving any new packets. 2) If the flow is active than max threshold time which is configurable. 3) If a TCP flag (FIN / RST) indicates the flow is terminated. NetFlow provides a powerful tool to understand the network behavior and is widely used for network monitoring. There are different tools that have similar flow approaches and a common standardization is done within the IETF IPFIX working group.

There are different variants of NetFlow with each advanced version giving additional information about the flow. The fields that we used in our experiment and that are supported across the versions are in figure 1.

2.1 Related Work

There has been plenty of attention paid in the area of Cyber Security to use ML/DM methods. Intrusion Detection Systems that help in discovering, determining and finding the unauthorized users are also embracing ML/DM techniques to handle sophisticated attacks which are being masqueraded by the humongous data flow.

Sperotto et al. [4] points out how traditional packet inspection cannot be used for high speed attacks detection and investigates alternate approach of flow-based intrusion detection. Netflow based network monitoring techniques are described in the work by Bahl et al. [1] Within the domain of networking a lot of relevant data has already been performed on processing Netflow data [3] [2]

3. DESIGN OVERVIEW

Detecting the behaviors implicit in the netflow data and using them to build systems that make network management easier is the overarching premise of this work.

We next describe the design of our system, and how the different components fit into the architecture.

3.1 Components

Our design is comprised of four essential components: Netflow Collector, Feature Engineering, Pattern Detector, and applications. These components are visible in the architecture diagram in figure 2.

3.1.1 Netflow Collector

The raw data collected at the routers is exported to Netflow collectors as described in section 2. NFDUMP tools are used to process the netflow data. They support netflow v5,v7 and v9. The two tools of primary importance are nfcapd (netflow capture daemon) that reads the netflow data from the network and stores the data into files. nfdump (netflow dump) reads the netflow data from the files stored by nfcapd. Using nfdump we import the data on to our local machines from the Netflow Collectors. This imported data forms the primary source for our experiments. We collect real network traffic from a campus network with almost XXX users.

3.1.2 Feature Engineering

Feature Engineering is the process of transforming raw data into features that better represent the underlying structure to the models so that they increase the accuracy of the model when applied on unseen data. This step comes before modeling and after seeing the data.

Features extracted from the data directly influence the models generated and predictions made out of it. The better the features are the better the results will be. The results that we achieve when we are using ML/DM approaches are a factor of the data set in hand, the features generated , the prediction model and the way the problem is framed. Most of the times even simpler models perform better if the features describe the structure inherent to the data. Feature Engineering encompasses of:

- Feature Scaling/Feature Normalization, a method used to standardize the range of independent features of data. Failure to standardize variables might result in algorithms placing undue significance to variables that are on a higher scale. There are different ways to do this namely, Min-Max scaling, Variance scaling, L2 nor-

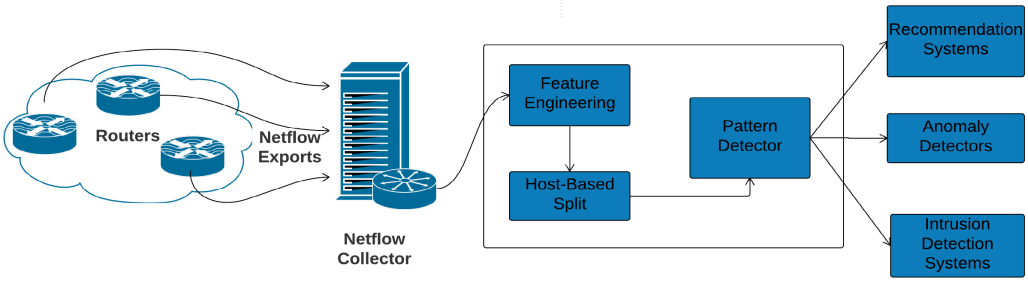

Figure 2: Architecture diagram

malization etc.. The choice depends on the data and different statistics of it such as Mean, Variance, L2 norm. Scaling is not advisable in all instances we might loose valuable information if applied without proper thought.

- Transforming Non-normal distribution to Normal, many ML/DM tools perform well on normalized data and having skewed data will give inaccurate results. Log transformations are generally used to normalize the skewed data. After transforming the data if it doesn't capture the essence of original data we could look at other options such as square root, cube root. BoxCox is also another techniaue that changes the distribution of variables from non-normal to normal or near normal.
- Missing data is a common issue that every data science experiment has to deal with. We applied rules such as consecutive rule (that says that there can be no missing values between the lowest and highest values for the attribute, and that all values must also be unique) and null rule (that specifies the use of blanks, question marks, or other strings that may indicate the null condition when a value for a given attribute is not available, and how such values should be handled.) to handle missing data.
- Features could be either numerical or categorical variables. When dealing with algorithms that work on numerical data we have to make sure that the categorical variables are handled. For example, in our case a category, Transport Protocol, has levels such as TCP, UDP and others. One approach is to encode them as 0, 1, and 2, respectively converting them to numerical variables. This could pose a problem when using distance measuring algorithms as the distance between the encoded attributes, like 1 (UDP) minus 0 (TCP) doesn't mean anything. Another approach would be to create a feature for each value of the categorical variable and mark it as 0 or 1 based on if the value is present or not. There are both pros and cons to this method. The pros are if we choose to aggregate the values on all features this conversion gives a numerical value to work with. The cons are scaling and standardization could be affected.
- Feature Selection refers to the process of selecting a subset of relevant features to model the data. It helps in shorter time periods of execution, overcoming the overfitting problem and making the solution more generic and avoid curse of dimensionality. The central premise of this process is to remove redundant or irrelevant features that don't make much sense to the problem we ought to solve. Feature selection and dimensionality reduction are two confusing terms. Though, both methods seek to reduce the number of attributes in the dataset, but dimensionality reduction does it by creating new combinations of attributes, where as feature selection methods include and exclude attributes present in the data without changing them. Principal Component Analysis, Singular Value Decomposition are examples of dimensionality reduction methods. The techniques used to do feature selection depend on text, numerical variables and they are three general classes of them Filter, Wrapper, Embedded methods. Filter methods assign a rank to each feature. They consider each feature independently, and all those features which are above certain rank are considered. Chi squared test, information gain and correlation coefficient are some example Filter Methods.

3.1.3 Host-Based Split

The main theme of our work is to learn more about behavior of the network. Packet based inspection could be detrimental for this and that is why we have chosen NetFlow as our source for investigation as it bundles packets into flows and alleviates the problem of looking into each packet. But, if we carefully examine a flow is an instance of communication between two hosts. There could be multiple such communications that could happen between the same hosts over the day and all these could be recorded as different flows as explained in the section 2. So, the netflow records of a day can consist of multiple entries for the same hosts communicating. Hence, to get the whole picture of a host we aggregate the data based

on source IP address. This is conducive in 2 ways. Firstly, With a small memory imprint we will be able to comment about behaviors of the host. Secondly, we deviate from the traditional approach of classifying based on applications or based on good/bad users.

3.1.4 Pattern Detector

3.1.5 Applications

4. IMPLEMENTATION

The design section talked about the different steps involved in our experiment in detail. Now let us look at how we implemented each of them and the choices that we have made at each step in pursuit of our goal.

We collected the netflow

5. REFERENCES

[1] BAHL, P., CHANDRA, R., GREENBERG, A., KANDULA, S., MALTZ, D. A., AND ZHANG, M. Towards highly reliable enterprise network services via inference of multi-level dependencies. In *ACM SIGCOMM Computer Communication Review* (2007), vol. 37, ACM, pp. 13–24.

[2] KARAGIANNIS, T., PAPAGIANNAKI, K., AND FALOUTSOS, M. Blinc: multilevel traffic classification in the dark. In *ACM SIGCOMM Computer Communication Review* (2005), vol. 35, ACM, pp. 229–240.

[3] LAKHINA, A., CROVELLA, M., AND DIOT, C. Mining anomalies using traffic feature distributions. In *ACM SIGCOMM Computer Communication Review* (2005), vol. 35, ACM, pp. 217–228.

[4] SPEROTTO, A., SCHAFFRATH, G., SADRE, R., MORARIU, C., PRAS, A., AND STILLER, B. An overview of ip flow-based intrusion detection. *IEEE Communications Surveys and Tutorials 12*, 3 (2010), 343–356.