Hierarchical Clustering Based Network Traffic Data Reduction for Improving Suspicious Flow Detection

Liya Su[1,2], Yepeng Yao[1,2], Ning Li[1], Junrong Liu[1], Zhigang Lu[1], Baoxu Liu[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Email: {suliya, yaoyepeng, lining6, liujunrong, luzhigang, liubaoxu}@iie.ac.cn

Abstract—**Attacks like APT have lasted for a long time which need suspicious flow detection on long-time data. However, the challenge of effectively analyzing massive data source for suspicious flow diagnosis is unmet yet. Consequently, flow data reduction should be adopted, which refers to abstract the most relevant information from the massive dataset. Existing approaches to sampling flow data are inherently inaccurate unless running at high sampling rate. In this paper, we proposed HCBS (Hierarchical Clustering Based Sampling), a flow data reduction scheme, to alleviate such problems. We study the characteristics of flow data relating malicious activities and employ hierarchical clustering to sample data for further deep detection. Experiments on 1999 DARPA dataset demonstrates that HCBS reduces the size of the flow data by 40% with only a small loss in accuracy and significantly outperforms the compared state-of-the-art.**

Keywords—flow data reduction, suspicious flow detection, hierarchical clustering, cluster sampling scheme

I. Introduction

With the rapid developments of Internet and computer systems, the importance of computer network security is growing with the widespread involvement of computers in daily life of human. As the wide use of malwares, network traffic contains all the communication information with the attackers [1] and provides the whole indicators of infection for discovering the malware sample activity[2]. Moreover, the extensive usage of highly dynamic network can provide serious threats, such as zero-day vulnerabilities, insider threats, etc. In order to ensure the protection and resilience of such networks, it is necessary to better analyze and observe network traffic. In principle, network traffic data contains a wealth of information about abnormal behavior that related to these malware samples and serious threats. As an important technology in defense-in-depth network security framework, detecting suspicious behavior has become a hot topic in computer networks in recent years [3][4].

The existing technologies for detecting suspicious behavior generally employ detection methods based on event analysis with preset patterns, namely misuse detection. The misuse detection is able to derive results with high precision (low number of false alerts), but their detection ability is limited only to the known samples and patterns included in the database (limited recall). Unlike misuse detection, suspicious flow detection is capable of identifying new threats and attempts to search for suspicious malicious behavior in network traffic data patterns, which deviate from established normal patterns. However, due to the continuous improvements of network bandwidth, analyzing all network traffic is intractable on high-speed network links. It is more efficient to classify network traffic based on flows representing groups of packets. While this approach has typically lower precision, it uses statistical modeling and behavioral analysis to find new and previously unforeseen threats with higher recall. Since unknown threats appear very frequently, pattern-based detection methods may be overwhelmed by an abundance of polymorphic threats. Thereafter, using network suspicious flow detection to discover unforeseen threats has become a necessity rather than an option.

This work is supported by Natural Science Foundation of China (No. 61702508), and National Key Research and Development Program of China (863 Program) (NO.2015AA017202). This work is also partially supported by Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences and Beijing Key Laboratory of Network security and Protection Technology.

Due to the huge volume of data and typically coming in a steaming fashion, prevent the malware communication in high-speed networks is really a challenging task. The existing suspicious flow detection techniques are limited to treat massive data effectively in big traffic environments. Although machine learning techniques, such as classification and clustering, have been utilized in suspicious flow detection systems to distinguish the normal from abnormal traffics, the efficiency needs to be improved. In order to address the huge volume and dynamic pattern of the traffic data, data reduction procedures with a near real-time performance is essential for effective suspicious flow detection approaches. Some commercial suspicious flow detection systems try to improve the efficiency by integrating sampling with detection algorithms in one framework. However, the drawback of these methods is that it fails to reduce with deriving desirable false negative rate and false positive rate.

Our goal in this paper is to take significant steps toward a system that fulfills these criteria simultaneously, namely, to reduce the normal data by analyzing subsets of network traffic with more deterministic and predictable traffic. We seek methods that can reduce a more precise set of traffic data without losing network anomalies, and to do so with high detection rate and low false alarm rate. We propose HCBS scheme aiming at building a hierarchical clustering based sampling method to reduce large-scale network traffic data in suspicious flow detection. Specifically, our proposed method is developed by cascading two machine learning algorithms: 1) the hierarchical clustering and 2) the Multinomial Naïve Bayes supervised learning model. In the first stage, hierarchical clustering is performed on network traffic flows to obtain disjoint clusters, with each cluster representing a region of instances. And then the Multinomial Naïve Bayes classifier is used to distinguish suspicious and normal instances for filtering them.

The main contributions of this work are summarized as follows:

- We analyze the impact of traffic sampling on the suspicious flow detection. The main limitation is that with the low sampling rate typically used by network operators (e.g., 1/ 1000) suspicious detectors could handle worst-case traffic scenarios and network attacks. We analyze this problem, both empirically and theoretical finding that clustering based sampling has a lower impact than other sampling methods on the performance of the suspicious detection.
- We employ Hierarchical Clustering Based Sampling (HCBS) scheme to extract data from the massive training data. We then build a lightweight cluster filter based on the smaller data set, which is an efficient system. We also prove that this process is effective for suspicious flow detection. To the best of our knowledge, this is the first work to reduce network traffic data for suspicious flow detection with machine learning approaches.
- We use http traffic datasets to evaluate the HCBS scheme for data reduction in suspicious flow detection. The extensive test results demonstrate the effectiveness of our method for network flow data reduction.

The rest of this paper is organized as follows. We present a review on prior research works on network traffic sampling and traffic data feature extraction and selection in Section II. Section III shows the preliminaries of the proposed approach. In Section IV, we describe a new scheme for network flow reduction based on hierarchical clustering. Section V illustrates performance evaluations of our proposed HCBS scheme on DARPA 1999 dataset. Finally, conclusions are drawn in Section VI.

II. Background and Related Work

Network suspicious flow detection has the ability to potentially detect unforeseen attacks, while suspicious flow detection techniques are therefore challenged by the demand to process more large-scale traffic data in higher dimensions at high speeds. A large amount of work is currently being performed in the field of intrusion detection. Most of the work focuses on improving the system's ability to identify suspicious network traffic and improving the network traffic's speed that can be handled.

This work is related to several topics in the area of http traffic and network data reduction for handling large volume of network traffic. At a high level, there are two classes of techniques regarding data reduction for suspicious flow detection for large-scale network. The first class of approaches relies on sampling data which is inherently inaccurate because potentially useful information can be discarded. The other class uses data mining and machine learning methods to perform for feature extraction and selection after the features have been constructed to reduce the dimensions of the originals.

A. Network Traffic Sampling

Network traffic sampling, originally proposed for network monitoring applications, now is used for reduce the analyzed data quantity for flow detection. In 1993, Kimberly C. Claffy et al.[9] first presented the performance of data sampling and compare packet-based with time-based sampling techniques answering data collection methods' importance related to wide area network traffic. However, network data sampling is considered an inherently lossy process which provides an fundamental bias that degrades the detection effectiveness of the underlying traffic in [11].

We review two well-known preferential flow-based sampling techniques. The first, referred to as selective sampling [8], targeting small flows (in terms of number of packets), follows the paradigm that small flows are usually the source of many network attacks. The second, smart sampling[9], is a type of flow based sampling that focuses on the selection of large flows. Both methods are based on hypothesis that the source of network attacks is usually small flows or large flows.

With packet sampling technique, Vilardi R et al. [12] proposed a lightweight enhanced monitored algorithm at router interfaces and Weizhi Meng et al. [13] applied into IoT(The Internet of Things) trust management. The effect of packet sampling on estimating traffic statistics has been well investigated [7]. These studies have shown that packet sampling has indeed an effect on the precision of estimating volume statistics. To solve this problem, Anukool Lakhina et al.[14] first analyzed the complete network traffic data and showed how to use PCA(Principal Component Analysis) to systematically decompose the structure of Origin-Destination flow time series. Fadlullah Z M et al. [16] and Salama A et al.[18] discussed using regression modelling, fuzzy inference system and deep learning to implement an adaptive sampling system responding to the traffic changes. However, ref [18] represented the original traffic only closely than the non-adaptive sampling method and [16] provided a comprehensive guide leaving more work to complete.

Our work proposed a new type of packet sampling, where malicious packets are sampled with higher probability, which improves the quality of anomaly detection.

B. Feature Dimensionality Extraction

Commonly there are two methods to decrease the dimensionality of the dataset, discarding redundant data and extract relevant features. Feature extraction deals with dimensionality of the data which transforms the initial feature set into a reduced number of new features.

Many researchers pointed out PCA [19] that is an unsupervised feature selection based on multivariate statistics and its basic idea is to seek a projection that represents the data in a best possible way in a least-square sense to provide dimensionality reduction. However, PCA is also known as Karhunen-Loeve transformation in pattern recognition is found not suitable in feature extraction in classification process for the non-inclusion of discriminatory information in calculating the optimal rotation of the feature axes.

As for time series reduction SVD is proposed by Abonyi[20] inspired by the sensor fusion algorithm. SVD gives a close approximation of time series data, whenever the observed variables exhibit a linear correlation. In other words, the SVD model is expected to give a high reconstruction error, if and only if the decomposed time series data fulfils a significant change in correlation structure.

As for the restriction of above methods, more flexible techniques are proposed. Chebrolu S. et al.[20] focuses on selecting major features through Bvayesian networks and classification and regression trees. Samant A et al.[22] presented a wavelet-based feature-extraction model for fuzzy neural network algorithm improving the traffic data incident-detection rate. Ahmad M Karimi et al.[23] proposed an IDS system architecture for efficient feature extraction using Netmap and Apache Spark. Yair Meidan et al.[24] proposed feature extraction on TCP packets for IoT traffic flows and with the specific several features the classifier could distinguish traffic from IoT and non-IoT devices. Niyaz Q et al.[25] use deep learning for feature selection of a large set of features derived from network traffic applying in software-defined network (SDN) environment.

We study the sampling problem using network flow data instead of packet-level traces. For example, NetFlow is a widely extended protocol developed by Cisco to export IP flow information from routers and switches. Using flow data has to limit the amount of information available to be used as features. Based on this basic information, additional features could be extracted, such as the average packet size or average inter-arrival time.

Different from network traffic sampling or feature dimensionality extraction, in this paper, we employ a scheme that finds benign clusters and suspicious clusters from original data and then analysis the clusters for filtering and further detection.

III. Preliminaries

In this section, we provide brief explanations of the concepts in traffic data reduction approach for large-scale network suspicious flow detection.

A. Error in Network Data Sampling

As for sampling network traffic, one of the main sources of inaccuracy under sampling is the estimation of the traffic features[26]. According to the paper, using sampling theory, the error in the estimation of the features presented in TABLE I. Although most of the features are fairly accurate for large flows and moderate sampling rates, some features are clearly biased. For those features that are unbiased, we find that the variance of the error is significant for small flows and low sampling rates.

The information available in NetFlow version five records is significantly limited leading to small number of features could be used. TABLE I. presents the set of features that can be used in general traffic data reduction. Based on the basic information, we compute some further features, such as the average packet size or average inter-arrival time, resulting in 14 features in total. TABLE I. also shows how each feature is computed according to the sampling rate *p* being applied.

One of the main error when sampling network traffic is the estimation of the traffic features[26]. Using sampling theory, in Section II, we could get the error in the feature estimation in TABLE I. . Most of the features are fairly accurate. However, some features are obviously biased, and the other unbiased features have significant error variance for small flows and low sampling rates, seen in (3). For example, with $p = 0.01$ and $n = 1$, the variance of the error in the number of packets is 1-0.01/0.01 = 99. It is the same for the flow size. Above all, we could expect the error of the classification results mainly from the flow features.

TABLE I. Set of Flow-based Features

Feature	Description	Value
sport	Source port of the flow	16bits
dport	Destination port of the flow	16bits
protocol	IP protocol value	8bits
ToS	Type of Service from the first packet	8bits
flags	Cumulative OR of TCP flags	6bits
duration	Duration of the flow in nsec precision	ts_{end}-ts_{ini}
packets	Total number of packets in the flow	$\frac{packets}{p}$
bytes	Flow length in bytes	$\frac{bytes}{p}$
pkt_size	Average packet size of the flow	$\frac{bytes}{packets}$
iat	Average packet inter-arrival time	$\frac{duration}{packets/p}$

TABLE II. Average Related Error of The Flow Features

Feature	P=0.5	P=0.1	P=0.05	P=0.01	P=0.005	P=0.001
sport	0.00	0.00	0.00	0.00	0.00	0.00
dport	0.00	0.00	0.00	0.00.	0.00	0.00
protocol	0.00	0.00	0.00	0.00	0.00	0.00
$\hat{f}$	0.05	0.16	0.18	0.22	0.23	0.24
$\hat{d}$	0.22	0.60	0.66	0.77	0.79	0.81
$\hat{n}$	0.66	3.66	6.90	29.69	55.17	234.61
$\hat{b}$	0.76	3.86	7.05	29.71	55.09	234.24
iat	0.29	0.65	0.71	0.78	0.80	0.82

$$\mathrm{Var}\left[1-\frac{\hat{n}}{n}\right]=Var\left[\frac{\hat{n}}{n}\right]=\frac{1}{n^2}Var[\hat{n}]=\frac{1}{n^2 p}n(1-p)=\frac{1-p}{np} \quad (3)$$

Every node in a network has two possible states, defined as H_1 and H_2. $P(H_1)$ is a possibility as in state H_1 in the absence of any other knowledge. In terms of obtained features as knowledge on the proposition H_1, record as features $F = \{f_1, f_2, f_3, ..., f_{10}\}$. And the belief would be the conditional probabilities as $p(H_1/F)$. Using the Bayes' theorem and assuming statistical independence between features: the posterior $p(H_1/F)$ can be calculated with $p(f_j/F)$ and prior $p(H_1)$.

$$\mathrm{p}(H_1/F)=\frac{\prod_{j=1}^{m}p(f_j/H_1)\cdot p(H_1)}{\sum_{i=1}^{2}\prod_{j=1}^{m}p(f_j/H_i)\cdot p(H_i)} \tag{4}$$

Given the error of every traffic features $E = \{e_1, e_2, e_3, ..., e_{10}\}$ and the statistical independence assuming, we could clearly get the error of random sampling. N is the number of the sampled network traffic flow.

$$E_{all}=\frac{1}{\mathrm{N}}\sum e_1\times N+\frac{1}{\mathrm{N}}\sum e_2\times N+...+\frac{1}{\mathrm{N}}\sum e_{10}\times N \tag{5}$$

B. Error in Hierarchical Clustering

We adopt hierarchical clustering method to cluster similar flows into the same cluster while dissimilar flows to different clusters. The average distance is used as linkage criterion which combines the distance of the corresponding flow features. With reference above, flow is decomposed into a set of features $F = \{f_1, f_2, f_3, ..., f_{10}\}$, the distance between F_i and F_j is define as:

$$\mathrm{dist}(\mathrm{F}i, Fj)=\sum_{k=1}^{n}Dist(f_k^i, f_k^j) \tag{6}$$

In order to reduce the computational burden of the error, a leader is elected for each cluster. Leaders will be the representative flows of their clusters. For example cluster N_i containing the flows $\{F_1, ...F_{Ni}\}$, the leader is the one that has minimum overall distance from the other members of the cluster, that is:

$$\arg\min_{Fi\in Ni}\left(\sum_{j=1}^{n}dist(F_i,F_j)\right) \tag{7}$$

In this paper we would discard most of flows in the clusters (for example 70%) that don't contain suspicious flows while maintain the whole clusters that contain suspicious flows. Given the error of every traffic features $E = \{e_1, e_2, e_3, ..., e_{10}\}$ and the statistical independence assuming, we could clearly get the error of hierarchical clustering, that is :

$$\begin{aligned}E_{HCBS} &= e_1\times N_s+e_2\times N_s+...+e_{10}\times N_s+\\ &e_1\times N_u\times 0.3+e_2\times N_u\times 0.3+...+e_{10}\times N_u\times 0.3\\ &=(\mathrm{e}1+e2+...+e10)\times(N_s+0.3\times N_u)<E_{all}\end{aligned} \tag{8}$$

N_s is the flow number of the suspicious clusters, N_u is the flow number of the normal clusters. According to the (8) and N_u is much larger than N_s, it's clear that the error of HCBS is much smaller than the error of other network data sampling approaches.

C. Supervised and unsupervised learning

Generally, machine learning approaches can be classified in unsupervised and supervised algorithms. Unsupervised learning algorithms try to find hidden structure in unlabeled data. Since the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. On the contrary, supervised machine learning algorithms learns from labeled instances or examples, which are collected in the past and represent past experiences in some real-world applications. They produce an inferred model, which can be then used for mapping or classifying new instances. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. Both supervised and unsupervised learning algorithms will be adopted in this paper.

We use unsupervised learning by applying a clustering algorithm called hierarchical clustering. Hierarchical clustering is a cluster analysis method which seeks to build a hierarchy of clusters. This clustering method has the distinct advantage that any valid measure of distance can be used. In fact, the observations themselves are not required: all that is used is a matrix of distances.

In the following we will use a type of hierarchical clustering that is called agglomerative: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. In order to decide which clusters should be combined, a metric (a measure of distance between pairs of observations) and a linkage criterion are required. In the following we will use a type of hierarchical clustering that is called agglomerative: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. Since we will cluster time-dependent sequences, we will use the total cost of an optimal warping path as distance metric. As for the linkage criterion, that determines the distance between sets of observations as a function of the pairwise distances between observations, we will use the average distance, that is defined as:

$$\mathrm{d}(u,v)=\sum_{\substack{1\le i\le n\\1\le j\le m}}\frac{d(u[i],v[j])}{|u|*|v|} \tag{9}$$

We use supervised learning by applying an ensemble classifier that is called Multinomial Naïve Bayes to filter the clustering results. Multinomial Naïve Bayes uses a simple, efficient, and effective discriminative parameter learning method. The discriminative parameter learning method learns parameters by discriminatively computing frequencies from network feature flows. From empirical studies we learn that the discriminative parameter learning has the advantages of both generative and discriminative learning techniques. In suspicious flow detection, it usually takes Bayesian networks as stable classifiers. Naïve Bayesian learning uses frequency estimate (FE) to classify data which compute the appropriate frequencies from data to determine parameters. FE as a typical generative learning method maximizes likelihood. What's important, it only needs to count each training instance once as its frequency which enables the classifier for high dimensional data very quickly. By comparison, another method ELR[27] outperforms the generative learning method FE, however, is limited to its high computational cost. So that, for efficient suspicious flow detection, discriminative parameter learning combining both generative and discriminative learning which maximizes generalization accuracy may be an obvious choice[28].

The procedure is specified as follows: (i) Estimate the current frequencies from the training data. (ii) Estimate the probabilities parameters. (iii) Compute the posterior probability for Multinomial Naïve Bayes model. (iv) Use the model to classify testing data.

IV. Hierarchical Clustering Based Sampling Scheme

In this Section, we present the scheme of hierarchical clustering based sampling designed for suspicious flow detection. Intuitively, the hierarchical clustering based sampling scheme should be a process in which number of samples and their distribution are selected in such a way that the loss of useful information is minimal.

In Fig. 1 we give an overview of the complete processing chain from packet capture to the generation of a suspicious flow detection result. We begin by describing flow generation procedure, calibrating network traces and extracting flow features both from normal and suspicious traffic. We then describe the hierarchical clustering algorithms, cluster analysis and filtering.

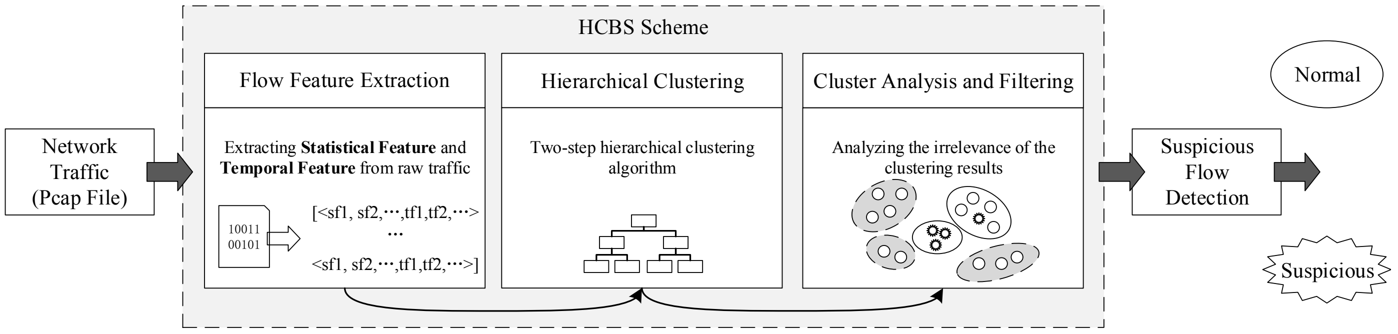

HCBS Scheme
Network Traffic (Pcap File)
Flow Feature Extraction
Extracting **Statistical Feature** and **Temporal Feature** from raw traffic
10011 00101
[<sf1, sf2,⋯,tf1,tf2,⋯>
⋯
<sf1, sf2,⋯,tf1,tf2,⋯>]
Hierarchical Clustering
Two-step hierarchical clustering algorithm
Cluster Analysis and Filtering
Analyzing the irrelevance of the clustering results
Suspicious Flow Detection
Normal
Suspicious

Fig. 1. Phases of the hierarchical clustering based sampling scheme

A. *Feature Extraction*

Raw traffic pre-processing is an important factor towards the initial composition of basic statistics regarding the network overall behavior and the individual host behavior. Our scheme relies on the statistics of network flows extracted from all sequence packets among hosts. The features of a flow that we use for data reduction are completely derived from the packet headers. These features describe the general behavior of a flow, for example, the size of transferred data in either direction, the packet size and inter-arrival time distributions.

Our scheme extracts features by grouping all flows by time intervals(eg. 10 minuts). Once the time interval has been derived, the following feature sets are extracted. The extracted features are defined as either structural features or temporal features. Flows in/out ratio is an example of structural features that characterize the regularity of flow behavior over time. Temporal features are likely to change as the flow progresses through time and reveal the variability of flows as a function of time.

- Structural Features: The first class of features extracted from flow data are based on flow sizes and packet numbers, which simply indicate the total number of bytes and total number of packets transferred in both in/out directions among all endpoints for a particular time interval. Our premise for analyzing structural flow features in flow data is that the statistical distributions for normal flows are significantly and necessarily different from statistical distributions for suspicious flows.
- Temporal Features: We selected temporal features that are dependent on time interval. In addition, the features do not include port-based, flag information (e.g., count of packets with the SYN flag set to 1 in the TCP option field) or payload-based analysis, which are vulnerable to the use of dynamic port numbers, encryption of payload, and masquerading techniques to mislead the suspicious flow detection algorithm. These vulnerable features may lead to optimistic detection performance. For example, using only port numbers yields accuracy similar to that obtained by using all the features in a supervised algorithm.

We select the structural flow features to model the behavior of traffic which have some minor elimination in the feature space proposed in previous section.

TABLE III. Two Categories of Extracted Features

Feature Category	
Structural Feature	***Temporal Feature***
Byte count	Bytes in/out ratio
Packet count	Packets in/out ratio
Flow count	Flows in/out ratio
Unique host count	Average bytes/packet
Unique source port count	Average packets/flow
Unique destination port count	Average bytes/flow
Unique destination country count	Average TCP session length

B. *Hierarchical Clustering Algorithm for Network Flows*

To attain high-quality clusters and decrease the computational cost of clustering, we adopt hierarchical clustering algorithm, as shown in Fig 2. The algorithm adopts a two-step clustering procedure to define the clusters. The two-step procedure was proposed to speed-up the process, by first using structural flow features to perform a coarse-grained clustering, and then by employing a set of temporal features to perform a fine-grained clustering. Both the coarse-grained and fine-grained clustering procedures are carried out by resorting to hierarchical clustering techniques, where data is aggregated in nested clusters and the clustering process terminates when further aggregation merges two distant clusters. Distance is measured by the Chebyshev norm between two time intervals.

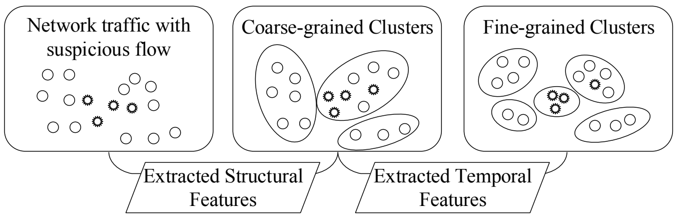

Network traffic with suspicious flow
Coarse-grained Clusters
Fine-grained Clusters
Extracted Structural Features
Extracted Temporal Features

Fig. 2. The two-step hierarchical clustering algorithm for network flows

1) Coarse-grained Clustering: In this step, we cluster flow time intervals based on structural features extracted from the network traffic flows. Therefore, computing the distance between pairs of time intervals reduces to compute the distance among vectors of numbers, which can be done efficiently.

2) Fine-grained Clustering: After splitting the network flow set into relatively large clusters by coarse-grained clustering, we further split each cluster into smaller groups. For this purpose, we consider each coarse-grained cluster as a separate set, measure the statistical similarity between the network flows in a cluster, and apply fine-grained clustering. This allows us to separate flow that have similar statistical characteristics (thus causing them to fall in the same coarse-grained cluster), but that present different temporal characteristics. Measuring the similarity between pairs of flow time intervals is relatively expensive. Since each coarse-grained cluster is much smaller than the total number of flow time intervals, fine-grained clustering can be done more efficiently than by applying it directly on the entire flow set.

The combination of coarse-grained and fine-grained clustering allows us to decrease the computational cost of the clustering process, compared to using only fine-grained clustering. This observations motivate the use of our two-step clustering process.

In both phases of our hierarchical clustering algorithm, we apply average-linkage hierarchical clustering. The main motivations for this choice are the fact that the hierarchical clustering algorithm is able to find clusters of arbitrary shapes, and can work on arbitrary metric spaces (i.e., it is not limited to distance in the Euclidean space). We ran pilot experiments using other clustering algorithms (e.g., complete-linkage hierarchical clustering). The average-linkage hierarchical clustering performed the best, according to our analysis. The hierarchical clustering algorithm takes a matrix of pair-wise distances among objects as input and produces a tree-like data structure where the leaves represent the original objects, and the length of the edges represent the distance between clusters.

C. *Cluster Analysis and Filtering*

The cluster analysis and filtering part compose a clustering engine that groups the flow features of all network flow into several clusters and classify clusters into suspicious flows or normal flows for filtering.

Clustering can be considered as an unsupervised learning task aiming at finding the cluster that produces the most compact and well separated clusters. No standard way exists of analyzing the irrelevance of a clustering result. Therefore, cluster analysis in this section involves the use of a subjective criterion of optimality by optimizing the Calinski-Harabaz index (a well-known cluster evaluation metric). We make use of the Calinski-Harabaz index in both the steps of the clustering process to automatically choose the best possible partitioning of the network flows.

After performing the clustering, the flow will be classified by the lightweight Multinomial Naïve Bayes classifier which was trained by a small quantity of training data. Multinomial Naïve Bayes is considered to be important for its simplicity, elegance and robustness. Note that Multinomial Naïve Bayes is an efficient and effective classification algorithm which assumes that all attributes are independent given the class (conditional independence assumption). But the attribute conditional assumption of Naïve Bayes rarely holds in real world applications. So, it needs to relax the assumption for classification.

The common belief is that the cluster center of each cluster presents the whole cluster better. We propose a sampling-like approach to analyze if the clusters are suspicious. We put *p%* nearest data to the cluster center into Multinomial Naïve Bayes classifier. If this part of cluster is classified as suspicious, the whole suspicious cluster would be input into the suspicious flow detection system. This approach is shown in Fig. 3. The way we use the *p%* data from cluster to present the whole cluster does well as shown in Section V.

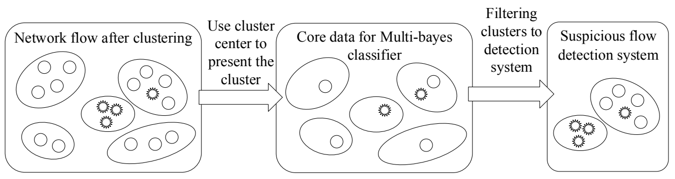

Network flow after clustering
Use cluster center to present the cluster
Core data for Multi-bayes classifier
Filtering clusters to detection system
Suspicious flow detection system

Fig. 3. The phrases of cluster analysis and filtering approach

The suspicious flow detection system could be an arbitrary classifier used on data that has been extracted features. We use various supervised classifiers include random forests, Majority Voting, and support vector machines (SVM).

The random forests algorithm not only achieves preferable detection rate but also greatly reduces training time using balanced data [33]. Moreover, the random forests are also able to detect some 'never-seen-before' anomalies.

Majority Voting has been used by several researchers utilizing the base classifiers to obtain better results [34]. There are some advantages in combining several classifiers such as increasing robustness, obtaining better accuracy, and heavily built generalization. The vote for one class is carried out by each base classifier, and the final class label is the one that receives more than half of the votes.

SVM is relatively a valid classification technique and has been shown higher performance than traditional learning methods in many applications. Ref[35] adopted SVM to network-based IDS and compared its performance to neural network-based IDS; the results show us that SVM gives better performance than neural network in terms of processing capacity and accuracy.

The main objective of suspicious flow detection system is to compare the performance between the reduction dataset and the original dataset, but not to compare the effect of the parameters for classifiers. As a result, for the sake of convenience, we just use the default values of the parameters for those classifiers.

Note that we only filter out widely known attack-free static flows to guarantee that no true attack is removed from the data. Comparing to other sampling methods, the data are not only largely reduced by this process, but also the suspicious data could be retained. That is because the filtering process use more information of data by the clustering method. With our method, it could believfe for the most time the flows about stealthy and continuous computer hacking process would reserve.

V. Evaluation

The goal of scheme evaluation is to verify the benefits of sampling technique on the effectiveness of various suspicious flow detection methods. The verification is based on comparison of different flow sampling techniques on network traffic data with potential threats (we did not focus on packet sampling methods due to the significantly worse results when compared to flow sampling[29] as well as on sampling techniques designed for specific suspicious flow detection methods which reduces their usefulness and applicability). In our evaluation, we performed three types of experiments by using three different sets of data. We measured performance of the sampling methods on traffic flow reduction, then we inspected the influence of the cluster analysis and filtering methods on traffic flow reduction and finally, we evaluated the impact of sampling on various suspicious flow detection methods. The reason of employing different sets of data is because of the nature of the experiments: network traffic from gigabit link is suitable for measuring sampling computational performance, but it is not very feasible to label such amount of network traffic to create the ground truth for evaluating the impact on anomaly detection.

A. Datasets

We choose 1999 DARPA dataset[5] as our experiment dataset. This is one of the most famous labeled intrusion detection dataset. It contains five weeks of raw traffic data, including both internal and external data. The data of the first three weeks are used for the training of proposed detection systems, consisting of two weeks of attack-free network data and one week of traffic data with labeled attacks. The fourth and fifth weeks of data are for testing. It contains millions of captured packets within a period of five weeks with four major categories of threats (i.e. R2L, U2R, DOS and Probe) in training set and some new type of threats (i.e. secret and sechole). In this dataset, there are some threats that do not affect the traffic patterns, so it can be used to effectively evaluate the generality of the proposed method.

In addition, the label files contain a few problems, such as duplicated records and incorrect labels. For example, the label file of the week 4 and week 5 contains a single record of "03/31/1999 18:29:25" with attack id 43.144547, which is an obvious timestamp error. Therefore, the dataset requires preprocessing before the experiments can be conducted. During week 5, the total 22 hours traffic data is available, and there is no downtime of the network. Due to this, we choose the first and second day in week 2 as training set for our cluster analysis and filtering method and the whole week 1, 3, 4 as testing set to evaluate out HCBS scheme.

The Nfcapd[29], Softflowd[31], Nfdump[32] tools were adopted to process the raw network traffic data into network flows. Second, every label file is checked, and all duplicated records and incorrect records should be removed. Finally, every network flow need be matched to the corrected labels.

B. Experimental Setup

In this paper, we implemented a proof-of-concept version of our hierarchical clustering based sampling scheme (see Section IV). Python3 and Sklearn0.19, which are run on the Ubuntu 16.04 64-bit OS, are used as the software frameworks. The server is a DELL R720 with 16 CPU cores and 16GB of memory.

We introduce two metrics to evaluate HCBS scheme, namely reduction rate and F-measure. The reduction rate is defined as N / N_T, where N_T is the total number of flows extracted by entire traffic, N is the number of flows after clustering and filtering. The F-measure used in this paper assigns the same weights to both Precision and Recall, and is given by:

$$F\text{ - }measure = \frac{2*(Precision*Recall)}{Precision+Recall}.$$

C. Results

We structured our experiments in three parts:

1) Analyze the effect of clustering and filtering parameters k (the number of cluster seen in this part) and p (the part of dataset for Multinomial Naïve Bayes seen in this part).

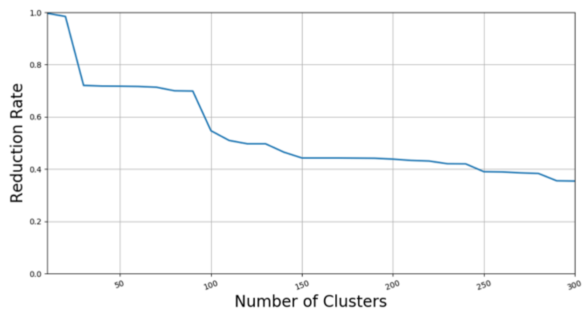
2) Evaluate the effectiveness of reduction the network traffic data using cluster analysis and filtering methods.

3) Evaluate the impact of HCBS for suspicious flow detection with various machine learning methods compared to random sampling and the whole dataset.

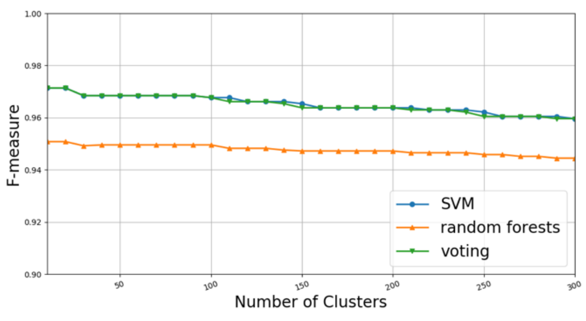
The 1999 DARPA dataset includes four weeks raw traffic data W_1, W_2, W_3, W_4 with labels. We treat W_2 as training data and the rest three weeks raw traffic data as training data. Note that W_2, W_4 contain attack records while W_1 and W_3 are attack free. We extracted features from four datasets and for convenience we call W_1, W_2, W_3 and W_4 as extracted features flows. We use small part of W_2 (denoted as T_{mnb}) as training data for Multinomial Naïve Bayes classifier, however for better suspicious flows detection result we use the whole W_2 as training data for suspicious flows detection system.

We enter W_1, W_3 and W_4 into Hierarchical Clustering system with N_c (the number of the clusters) clusters output. We selected *p%* (as the parameter p) of core data from every cluster into Multinomial Naïve Bayes classifier. We'll see parameter p is a small number in the back for Multinomial Naïve Bayes is a lightweight classifier. If we find at least one suspicious flow in every cluster, we label this cluster as suspicious cluster. If not, we filter this cluster from the further process. After filtering, we import the rest data into the suspicious flow detection system and get the classification result as P, R and F.

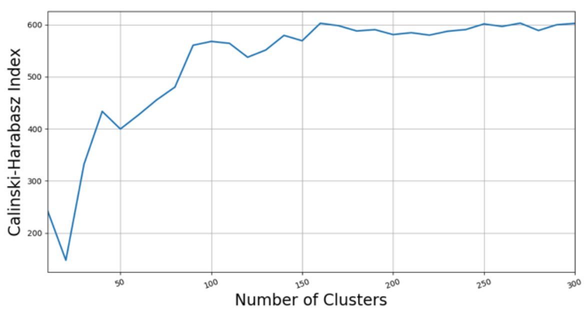
Flow Reduction Results To compare the flow reduction results of the HCBS scheme with the proposed clustering method and the lightweight cluster filtering model, we use two feature category and the general suspicious detection algorithm. Fig. 4(a) and Fig. 4(b) illustrate the results that the data reduction rate and F-measure both decrease with a larger number of clusters. While F-measure decrease much slower, the affection can be neglected. Fig. 4(c) illustrate that when the number of clusters is larger than 150, the corresponding Calinski-Harabaz index reaches to around 600.



(a)



(b)



(c)

Fig. 4. Flow reduction results. (a)Flow reduction rate over number of clusters. (b)Detection accuracy of suspicious flow over number of clusters. (c) Calinski-Harabaz index over number of clusters.

Evaluation of Lightweight Cluster Filter To remove normal clusters from the clusters generated by the two-step clustering, we use different training sets to construct a lightweight cluster filter and evaluate the effectiveness of the functionality. Experiment shows that the sampling rate of each cluster that is chosen by light weight cluster filter has little effect on accuracy and efficiency. Fig. 5 reports the achieved results that for high sampling rate (e.g., p = 0.1), the F-measure is larger than that for low sampling rate (e.g., p = 0.001).

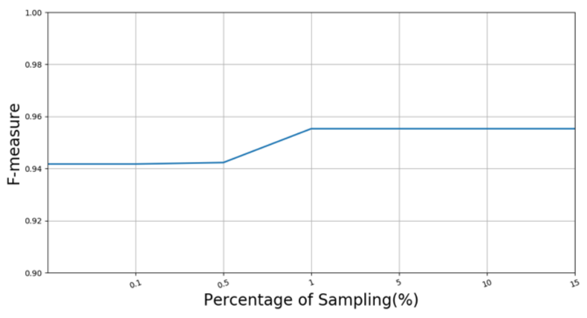


Fig. 5. Lightweight Cluster Filter with different training sets

Impact on Suspicious Flow Detection To illustrate the impact of accuracy and efficiency on suspicious flow detection, we use a general network flow classifier for results comparison. We classify the dataset into two main classes: normal and suspicious. In order to evaluate the proposed scheme, we use support vector machines (SVM) to train a binary classifier. Experiment has been conducted to evaluate the performance of the proposed scheme. Fig. 6 shows the ROC area for data with and without sampled. The sampled flow data have ROC area of 0.82 which shows a better result than that not sampled flow data.

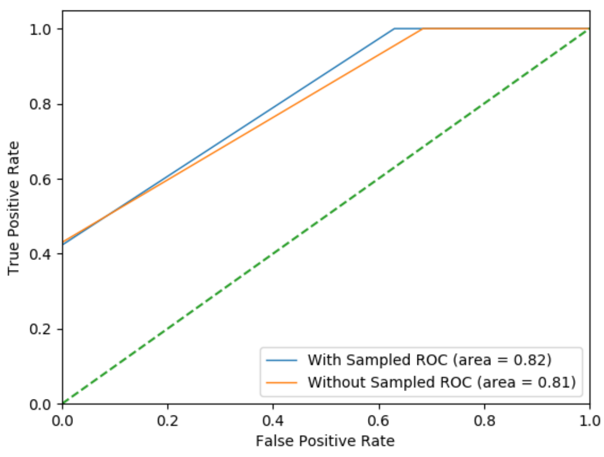


Fig. 6. Impact on three types of suspicious flow detection approach

Compared to Other Sampling Methods To compare the impact of accuracy and efficiency with other sampling methods, we choose selective sampling and smart sampling to evaluate the ability of preserving overall accuracy. The impact of above-mentioned sampling methods on suspicious data preserving and evaluation metrics is described in TABLE IV. In our evaluation, we clearly demonstrated that the HCBS are much more reversible than the selective sampling and smart sampling.

VI. Conclusions

The amount of data in suspicious flow detection is becoming increasingly massive in current large-scale network environments. In this paper, we addressed the traffic data reduction problem with flow data using a novel scheme which employed both supervised and unsupervised learning methods. Our results allow us to come to the conclusion that: our network data reduction scheme HCBS is very effective that can reduces the size of the flow data with only a small loss in accuracy. This scheme improves detection efficiency for two reasons: first, only a smaller set of data needs to be further processed for the backend suspicious flow detection system, and second, the data reduction process only needs to be based on a lightweight detection model. Experimental results on well-known dataset showed that HCBS can accurately and efficiently reduce normal flows from massive network traffic simultaneously.

TABLE IV. Comparation of Three Sampling Methods

Suspicious Detection Approach	**Sampling Method**	**Performance over around 40% Time Interval Data Reduction**				
		Normal Time Intervals	***Suspicious Time Intervals***	***Precision***	***Recall***	***F-measure***
SVM	None	1955	86	0.954568	0.961783	0.956549
	Selective Sampling[8]	1189	37	0.973979	0.964111	0.968056
	Smart Sampling[9]	1153	70	0.935228	0.947670	0.934552
	HCBS	1042	74	0.925466	0.936380	0.929064
Random Forests	None	1955	86	0.976579	0.975992	0.971359
	Selective Sampling[8]	1189	37	1.000000	1.000000	1.000000
	Smart Sampling[9]	1153	70	0.961568	0.959935	0.949565
	HCBS	1042	74	0.963824	0.962366	0.955281
Voting	None	1955	86	0.976579	0.975992	0.971359
	Selective Sampling[8]	1189	37	1.000000	1.000000	1.000000
	Smart Sampling[9]	1153	70	0.961568	0.959935	0.949565
	HCBS	1042	74	0.963824	0.963824	0.963824

Acknowledgments

The authors would like to thank the anonymous referees for their valuable comments and corrections. This research was supported by the Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences and Beijing Key Laboratory of Network Security and Protection Technology(NO: 2015AA017202 and 61702508).

References

[1] Perdisci R, Lee W, Feamster N. Behavioral Clustering of HTTP-Based Malware and Signature Generation Using Malicious Network Traces[C]//NSDI. 2010, 10: 14.

[2] Lever C, Kotzias P, Balzarotti D, et al. A Lustrum of malware network communication: Evolution and insights[C]//Security and Privacy (SP), 2017 IEEE Symposium on. IEEE, 2017: 788-804.

[3] Ha, T., Kim, S., An, N., Narantuya, J., Jeong, C., Kim, J., & Lim, H. (2016). Suspicious traffic sampling for intrusion detection in software-defined networks. Computer Networks, 109, 172-182.

[4] Celik, Z. B., Walls, R. J., McDaniel, P., & Swami, A. (2015, October). Malware traffic detection using tamper resistant features. In Military Communications Conference, MILCOM 2015-2015 IEEE (pp. 330-335). IEEE.

[5] Lippmann R, Haines J W, Fried D J, et al. The 1999 DARPA off-line intrusion detection evaluation[J]. Computer networks, 2000, 34(4): 579-595.

[6] Shiravi, A., Shiravi, H., Tavallaee, M., & Ghorbani, A. A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. computers & security, 31(3), 357-374.

[7] Androulidakis, G., Chatzigiannakis, V., & Papavassiliou, S. (2009). Network anomaly detection and classification via opportunistic sampling. IEEE network, 23(1), 6-12.

[8] Androulidakis, G., & Papavassiliou, S. (2008). Improving network anomaly detection via selective flow-based sampling. IET communications, 2(3), 399-409.

[9] Duffield N, Lund C. Predicting resource usage and estimation accuracy in an IP flow measurement collection infrastructure[C]//Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement. ACM, 2003: 179-191.

[10] Claffy K C, Polyzos G C, Braun H W. Application of sampling methodologies to network traffic characterization[C]//ACM SIGCOMM Computer Communication Review. ACM, 1993, 23(4): 194-203.

[11] Ali S, Haq I U, Rizvi S, et al. On mitigating sampling-induced accuracy loss in traffic anomaly detection systems[J]. ACM SIGCOMM Computer Communication Review, 2010, 40(3): 4-16.

[12] Vilardi R, Grieco L A, Barakat C, et al. Lightweight enhanced monitoring for high speed networks[J]. Transactions on Emerging Telecommunications Technologies, 2014, 25(11): 1095-1113.

[13] Meng W, Li W, Su C, et al. Enhancing Trust Management for Wireless Intrusion Detection via Traffic Sampling in the Era of Big Data[J]. Ieee Access, 2018, 6: 7234-7243.

[14] Lakhina A, Papagiannaki K, Crovella M, et al. Structural analysis of network traffic flows[C]//ACM SIGMETRICS Performance evaluation review. ACM, 2004, 32(1): 61-72.

[15] Duffield N, Lund C, Thorup M. Estimating flow distributions from sampled flow statistics[C]//Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications. ACM, 2003: 325-336.

[16] Hohn N, Veitch D. Inverting sampled traffic[C]//Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement. ACM, 2003: 222-233.

[17] Fadlullah Z M, Tang F, Mao B, et al. State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems[J]. IEEE Communications Surveys & Tutorials, 2017, 19(4): 2432-2455.

[18] Salama A, Saatchi R, Burke D. Adaptive sampling technique using regression modelling and fuzzy inference system for network traffic[J]. 2017.

[19] Fukunaga K, Koontz W L G. Application of the Karhunen-Loeve expansion to feature selection and ordering[J]. IEEE Transactions on computers, 1970, 100(4): 311-318.

[20] Abonyi J, Feil B, Nemeth S, et al. Principal component analysis based time series segmentation: A new sensor fusion algorithm[J]. preprint, 2004.

[21] Chebrolu S, Abraham A, Thomas J P. Feature deduction and ensemble design of intrusion detection systems[J]. Computers & security, 2005, 24(4): 295-307.

[22] Samant A, Adeli H. Enhancing Neural Network Traffic Incident‐Detection Algorithms Using Wavelets[J]. Computer‐Aided Civil and Infrastructure Engineering, 2001, 16(4): 239-245.

[23] Karimi A M, Niyaz Q, Sun W, et al. Distributed network traffic feature extraction for a real-time IDS[C]//Electro Information Technology (EIT), 2016 IEEE International Conference on. IEEE, 2016: 0522-0526.

[24] Meidan Y, Bohadana M, Shabtai A, et al. ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis[C]//Proceedings of the Symposium on Applied Computing. ACM, 2017: 506-509.

[25] Niyaz Q, Sun W, Javaid A Y. A deep learning based DDoS detection system in software-defined networking (SDN)[J]. arXiv preprint arXiv:1611.07400, Add to Citavi project by ArXiv ID 2016.

[26] Carela-Español V, Barlet-Ros P, Cabellos-Aparicio A, et al. Analysis of the impact of sampling on NetFlow traffic classification[J]. Computer Networks, 2011, 55(5): 1083-1099.

[27] Grossman D, Domingos P. Learning Bayesian network classifiers by maximizing conditional likelihood[C]//Proceedings of the twenty-first international conference on Machine learning. ACM, 2004: 46.

[28] Panda M, Abraham A, Patra M R. Discriminative multinomial naive bayes for network intrusion detection[C]//Information Assurance and Security (IAS), 2010 Sixth International Conference on. IEEE, 2010: 5-10.

[29] Mai J, Chuah C N, Sridharan A, et al. Is sampled data sufficient for anomaly detection?[C]//Proceedings of the 6th ACM SIGCOMM conference on Internet measurement. ACM, 2006: 165-176.

[30] Nfcapd. https://github.com/YasuhiroABE/ansible-nfcapd

[31] Softflowd. https://github.com/davidediger/softflowd

[32] Nfdump. https://github.com/phaag/nfdump

[33] Zhang J, Zulkernine M, Haque A. Random-forests-based network intrusion detection systems[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2008, 38(5): 649-659.

[34] Kachirski O, Guha R. Effective intrusion detection using multiple sensors in wireless ad hoc networks[C]//System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on. IEEE, 2003: 8 pp.

[35] Mukkamala S, Janoski G, Sung A. Intrusion detection using neural networks and support vector machines[C]//Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on. IEEE, 2002, 2: 1702-1707.