A Scheme for Building A Dataset for Intrusion Detection Systems

Van Loi Cao
Faculty of information technology
Le Quy Don Technical University
Hanoi, Vietnam
loicv79@gmail.com

Van Thuy Hoang
Faculty of information technology
Le Quy Don Technical University
Hanoi, Vietnam
hoangvanthuy90@gmail.com

Quang Uy Nguyen
Faculty of information technology
Le Quy Don Technical University
Hanoi, Vietnam
quanguyhn@gmail.com

Abstract—**One of the main challenges in developing a network-based intrusion detection system is collecting data for training the system. Although, some datasets such as KDD Cup 1999 have been collected and are in public, these datasets are out of date and unreliable for building a system in reality. In this paper, we propose a scheme for building online an intrusion detection dataset. The scheme allows us to collect the raw data from a controlled environment and then process to have 16 features (traffic and content features) with full labels. The collected dataset is called LUT13. We then applied two well-known machine learning techniques: Artificial Neural Network (ANN) and Fuzzy C-Means (FCM) to train the system based on this dataset. The system, after trained on LUT13, was tested on the real environment and compared with the system constructed based on KDD Cup 1999. The results show that our dataset helps the system achieves higher detection rate compared to KDD Cup 1999.**

Keywords-intrusion detection systems; fuzzy c-means (FCM); artificial neural network (ANN); KDD Cup 1999; real-time intrusive dataset;

I. Introduction

Network-based intrusion detection systems (NIDSs) have played an essential role in network security due to the widespread use of computer network, the increase in valuable resources and the rapid development of attackers [1, 2]. A NIDS usually observes network behavior by analyzing the content (e.g., payload) and statistical features of network traffic (e.g., header of IP, TCP or ICMP and information of connections). In general, network security practitioners have to perform two major stages to develop a NIDS. The first stage is applying a machine learning algorithm to distinguish normal and intrusive activities. The goal is to raise the detective speed and minimize false alarm rate of intrusion detection. The second stage aims to evaluate these techniques on real world data. However, collecting such testing data is one of the main challenges in intrusion detection research. The reason is due to the lack of network traffic or host logs and events. Moreover, gathering data from real environment may raise the private and secure concerns [3, 4].

In 1998, DARPA Intrusion Detection Evaluation program (IDEVAL) [5] was the first attempt to evaluate intrusion detection techniques at MIT Lincoln Laboratory. The traffic was generated on a simulated network which was similar to the Air Force network. The training data was collected within seven weeks and covered twenty-four attack types. The testing data was built from another two weeks and contained fourteen new attack types. It was then made public with format of tcpdump files. Lately, Lee et al. [1, 2] constructed 41 attributes from the IDEVAL data mentioned above to built classification models for network based intrusion detection. The raw traffic dump from the IDEVAL was converted into unique connection records. The TCP dump data from seven weeks was processed into approximately five million connection records as the training data while the two million connection records of testing data was converted from the additional two weeks. The datasets were used as KDD Cup 1999 data (The 1999 Knowledge Discovery and Data Mining Tools Competition) [6]. The model could classify connection records into good (e.g., normal) connections and intrusive activities. All attacks fell into four main categories as Denial of Service (DOS), unauthorized access from a remote machine (R2L), unauthorized access to local superuser privileges (U2R) and surveillance and other probing (Probe).

Over the last decade, both of the DARPA datasets and KDD Cup datasets have become increasingly popular among the network security community. Some researchers used the DARPA datasets to evaluate their systems, whereas the KDD Cup datasets were used by others who developed machine learning techniques for intrusion detection systems [4]. This data set provides a common platform for NIDS evaluation, meaning that the results of current techniques can be compared with those of the previously published results [3, 4].

However, the KDD Cup datasets have faced some criticisms in recent years. Sabhani and Serpen [7] recommended that machine learning algorithms should not be applied for the two rare classes in KDD Cup datasets (e.g., R2L and U2R) in misuse detection context because of the different of target hypotheses in training and test data sets. Muda et al. [8] also pointed out that their hybrid of K-Means clustering and Naive Bayes detected these two classes inefficiently. Moreover, due to the rapid changes in networking technologies and computation, and the evolvement of attacks, KDD Cup datasets have been old-fashioned. Therefore, they are unreliable for building a real NIDS. Besides, the attributes of KDD Cup datasets might be problematic because of some unnecessary features and even noise [9]. This would decrease the detection accuracy rate and processing speed while increase system's workload and false alarm rate.

In this paper, we propose a scheme for building an online dataset, namely LUT13 dataset, for NIDSs. The dataset follows the format of 16 features which were chosen from 41 features in KDD Cup datasets [9]. We then evaluated the dataset with FCM and ANN under the same environment. The results demonstrate that the KDD Cup datasets are out of date and achieve lower detection rate compared to our dataset.

The rest of the paper is organized as follows. Section 2, we briefly review related work. In Section 3, the scheme for collecting an intrusion detection dataset in a real-time environment is discussed in detail. The experiments and results are presented in Section 4. Section 5 concludes the paper and highlights some future work.

II. RELATED WORK

Developing an effective NIDS is one of the main strands in network security. Recently, researchers have proposed a number of approaches in this field. The techniques can be categorized into three main groups: finding relevant or important features; improving machine learning algorithms and evaluating techniques on real-time dataset. Some of the techniques are detailed follows.

Yang Li et al. [10] presented a new lightweight intrusion detection model which first selected some important features by using Information Gain (IG) and Chi-Square techniques. These features were then experimented on KDD Cup datasets. Similarly, IG and Wrapper with Bayesian Networks (BN) and Decision trees (C4.5) methods were applied to select subsets of features on KDD Cup. The result showed that using a subset of key attributes for each of four categories DOS, Probe, R2L and U2R still maintained accuracy. Moreover, using a smaller set of features could reduce the computational cost of NIDS considerably [9].

Some other researchers focused on the capability of their techniques in distinguishing normal activities from anomalies and the ability to detect new or unknown attacks. Panda et al. [11] applied K-Means and Fuzzy C-Means in detecting anomaly based network intrusions, and they proved that these algorithms were good enough in terms of accuracy and time-consuming. Muda et al. [8] used a supervised learning technique, Naïve Bayes, to divide dataset into the normal instances and potential attacks. Attacks group was then classified into the four specific categories by using K-Mean algorithm. Although, the detection rate was improved to 99.6%, there was a limitation on detecting R2L and U2R attacks.

Apart from analyzing offline KDD Cup dataset, recently, some researchers have applied machine learning algorithms on their real-time systems and achieved considerable successes. Komviriyavut et al. [12] presented an approach to build dataset of 13 features, namely, RLD09 and used decision tree algorithm and Ripper rules to categorize them into three types DOS, Probe and normal. The detection rates were over 98% with a reasonable detection speed and the NIDS techniques also had a potential capability to differentiate new or unknown attacks. However, RLD09 did not have any payload features that often associated with DOS, R2L and U2R [9]. After that, Kamran Shafi et al. [3, 4] proposed a better real-time approach to generate a full label network intrusion detection dataset, namely, Seal dataset which were appropriate to both supervised and unsupervised machine learning algorithms. Most of the 33 features in the Seal dataset are payload based, which not only detected intrusive actions efficiently, but was also able to warn viruses and spywares. However, the evaluation of their systems was not tested in a real environment due to security and privacy concerns. Recently, Tran et al. [13] integrated block-based neural network (BBNN) which was improved with a high-frequency FPGA circuit to design a real-time intrusion detection system. The most interesting result is the high speed of the system, which made it suitable for processing large-scale datasets from real-time mode. However, the system depended heavily on Netflow technology. Therefore, it is difficult to modify the feature set for each environment.

Most of the above techniques focused on evaluating machine learning methods on the publicly available datasets known as KDD Cup datasets, which have been out of date. There were only two systems which were constructed based on their own feature vectors. However, their features were not clearly discussed. Therefore, it is very difficult implement their methods when building a real NIDS. For this reason, it is necessary to generate a KDD Cup-based dataset, which is up to date and simple to collect when we construct an Intrusion Detection System.

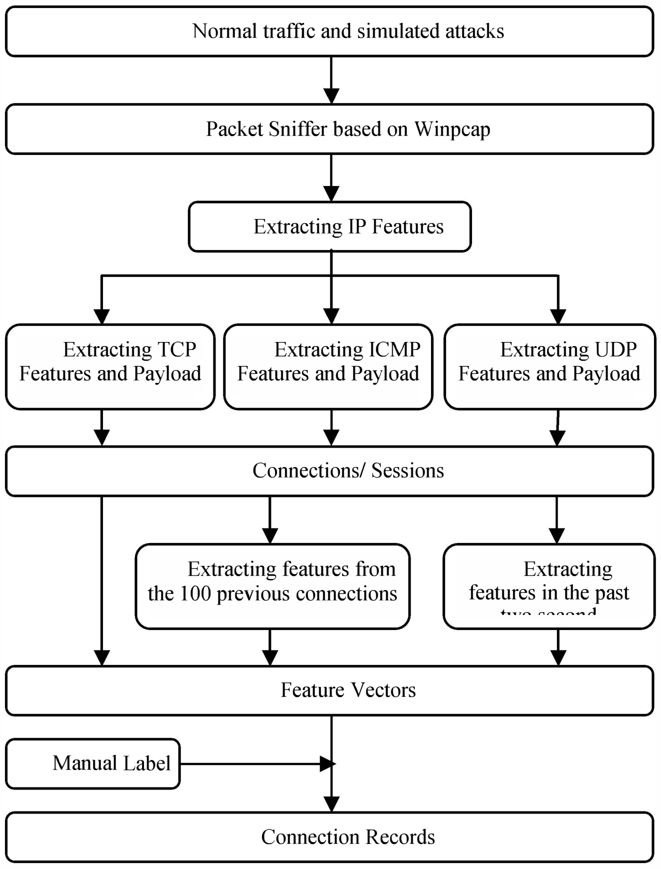
Normal traffic and simulated attacks
Packet Sniffer based on Winpcap
Extracting IP Features
Extracting TCP Features and Payload
Extracting ICMP Features and Payload
Extracting UDP Features and Payload
Connections/ Sessions
Extracting features from the 100 previous connections
Extracting features in the past
Feature Vectors
Manual Label
Connection Records

Figure 1. The scheme for collecting dataset from real-time traffic

III. A REAL-TIME SCHEME TO EXTRACT FEATURES

The scheme for collecting LUT13 dataset is summarized in Fig 1. The main steps include generating real-time traffic, extracting features and combining all data into connection records with labels. Firstly, Packet Sniffer captures raw traffic which is produced by background traffic and simulated attack sessions. Both protocol and payload features which are extracted from traffic dump are combined together to create connections or sessions. Following this, feature vectors are produced by putting attributes of not only current connections but also connections within the past two seconds and the 100 previous connections. The final step is to generate connection records by labeling manually. The output of the process is a dataset including 16 features as in [9]. Each of the steps is discussed in detail in the following sections.

A. *Generation traffic and simulation of attacks*

The first step to collect the dataset is generating a network traffic. The traffic must consist of both normal activities and intrusive actions. However, due to privacy and security of the specific networks, gathering data from these sites may not be allowed or is restricted to publicity [1]. Therefore, we designed a local network in the laboratory in our university. The network is very similar to the DARPA's network and has the inside and outside part. However it consists of a smaller number of network services.

The two processes of generation background traffic and anomalous traffic are separated completely. However, each illegitimate activity also contains normal activities, which makes our data closer to the real environment. The session of creating pure traffic is also necessary since it exists in the normal situations. In order to simulate patterns of illegitimate actions, we use the publicly available attack tools that are kept up to date.

B. *Winpcap-based Packet Sniffer*

Most networking applications access the network through widely used operating system primitives such as sockets. It is easy to gather data from the network because the operating system deals with the low level details (protocol handling, packet reassembly, etc.) and provides a familiar interface. However, this approach does not always satisfy, since some applications (analysis, security, monitoring, etc.) require directed access to packets on the network. In such situations, the "raw" data without any interposition of protocol processing by the operating system needs to provide [14].

WinPcap is an open source library for packet capture and network analysis on the Win32 platforms [14]. Its purpose is to give this kind of access to Win32 applications. In this research, we aim to design a Packet Sniffer Tool which gathers raw packets and converts them into the format of tcpdump by using WinPcap API in C# environment. The architecture of WinPcap is described more details in [15]. The function of parsing is supported, which makes our Packet Sniffer to be able to analyze both packet headers and payload (content of http, ftp, etc.).

C. *Extracting features*

In the KDD Cup datasets, a connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocols [2]. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record is a labeled feature vector consisting of forty-one features which category into four groups (basic, payload, time-based traffic and host-base traffic feature) [6].

In our research, we choose only a 16-feature vector from 41 features of KDD Cup. There are two reasons for this selection. Firstly, this helps to reduce computational cost of NIDS and remove redundant features [9]. Secondly, the unbalance between the number of DOS, Probe records and R2L, U2R records in KDD99 dataset results in the inefficiency of machine learning algorithms [7, 8]. Hence, we restricted our research to two most popular groups of attacks (DOS and Probe). These 16 features are described in four tables below.

1) Basic features of individual TCP connections: Basic features can be derived from packet headers without inspecting the payload. They are 5 features listed in Table I.

TABLE I. BASIC FEATURES

Num	Name	Description
1	Service	Destination service (e.g. telnet, ftp)
2	Flag	Status flag of the connection
3	src_bytes	Bytes sent from source to destination
4	dst_bytes	Bytes sent from destination to source
5	Wrong_fragment	Number of wrong fragments

2) Content features: Domain knowledge is used to assess the payload of the original TCP packets. This includes features in Table II.

TABLE II. PAYLOAD FEATURES

Num	Name	Description
6	Hot	Number of "hot" indicators
7	num_compromised	Number of "compromised" conditions

3) Time-based traffic features: The time-based traffic features are constructed particularly to detect high volume fast rate DOS attacks based on the number of connections made to the same destination host or service in the past two seconds.

TABLE III. TIME-BASED TRAFFIC FEATURES

Num	Name	Description
8	dst_host_count	Count of connections having the same destination host
9	dst_host_diff_srv_rate	% of different services on the current host
10	dst_host_srv_diff_host_rate	% of connections to the same service coming from different hosts

11	dst_host_srv_serror_rate	% of connections to the current host and specified service that have an S0, S1,S2,S3 error
12	dst_host_rerror_rate	% of connections to the current host that have an RST error

4) Host-based Traffic Features: In KDD99 dataset, some probing attacks scan the hosts (or ports) using a much larger time interval than two seconds. Therefore, connection records were also sorted by destination host, and features were constructed using a window of 100 connections to the same host instead of a time window. This yields a set of so-called host-based traffic features.

TABLE IV. HOST-BASED TRAFFIC FEATURES

Num	Name	Description
13	Count	Number of connections to the same host as the current connection in the past two seconds
14	srv_count	Number of connections to the same service as the current connection in the past two seconds
15	same_srv_rate	% of connections to the same service
16	diff_srv_rate	% of connections to different services

D. Labelling

Finally, each feature vector is labeled as either normal or intrusive connection, with exactly one specific attack type. This is based on the documentation of Lincoln Laboratory's website [16]. Following this, each attack is simulated in the different period of time to specific destinations in our network. This provides some types of necessary information such as attack name, starting/ending time, victim, etc which helps us to produce connection records with exact labels.

IV. EXPERIMENTS AND RESULTS

This section presents the experiments of using some well-known machine learning techniques on our dataset. Machine learning, a branch of artificial intelligence, studies the construction of systems that can learn from data. Since it was introduced in early 1960s, machine learning has greatly developed leading to the emergence of a number of powerful techniques. Among them, artificial neural networks (ANN) and Fuzzy C-Means (FCM) have been successfully used in many applications and are used in the experiments in this paper.

A. Experimental settings

Our physical network is depicted in Fig 2. This network is very similar to DARPA's network, but consisting of a smaller number of network services, users and clients. The network includes an inside and an outside component separated by a router. The outside part includes a web server and two workstations which were used to simulate attack sessions to inside. There are some workstations and three servers running Web, Mail, Ftp, DNS, DHCP, Directory and another IDS server. IDS server captures background traffic and flows to inside victims from inside and outside parts.

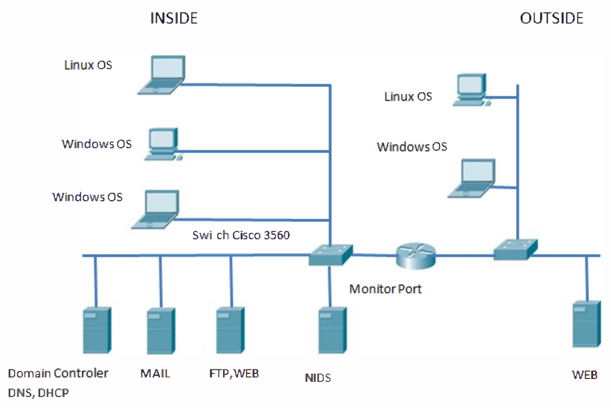

Figure 2. Diagram of the physical network in the experiment

In order to generate the normal traffic, we set up basic services as HTTP, DNS, DHCP, FTP, SMTP and NETBIOS on the network. Many publicly available hacking tools are used to simulate attack sessions to inside victims. Most attack tools are used in training phase and we add three others for testing phase. These tools are presented in Table V.

TABLE V. TRAINING AND TESTING HACKING TOOLS

Hacking tool	Attack name	Train	Test
Net tools 5	HTTP flood, UDP flood, Advanced Port Scan	√	√
Jping.c	Jping	√	√
Smurf.c	Smurf	√	√
Targa2.c	Land, Teardrop, Nestea, Newtear	√	√
Nmap	IP scan, ACK scan, FIN Stealth, UDP scan, Connection scan	√	√
SuperScan4	TCP scan	-	√
Targa2.c	SYN drop	-	√
Nmap	SYS Stealth	-	√

We employed a group of students to use the network as real activities and simulate the hacking tools during the seven-day period. Each attack type and normal activities was collected and converted into connection records separately, which helps us label accurately. These connection records were then mixed together to produce LUT13 dataset.

In order to generate the training data, we used a group of five hacking tools to attack inside victims during the four-day period and background traffic is generated for another day. The training dataset consists of 20,000 records (7000 DOS records, 7000 Probe records and 6000 normal records) with 14 attack types. For the last two days, we collected 8000 connection records for the testing dataset (3000 DOS records, 3000 Probe records and 2000 normal records) with the 14 attack types and 3 unknown attack types (the attacks that are not used in the training data).

B. Experimental Results

To demonstrate the efficiency of our dataset in building a NIDS, we conducted an experiment as follows. Firstly, we trained ANN and FCM on LUT13. After that, ANN and FCM were also trained on KDD Cup. Four systems, obtained after training process, were tested on the real environment. We measured the detection rate of these systems and the results are showed in Table VI.

It can be seen from this table that both ANN and FCM achieve greater detection rate when they are trained on LUT13 than on KDD Cup. These results evidence that KDD Cup datasets have been obsolete and only suitable for comparing between different algorithms. When the objective is to build a real NIDS, it is necessary that network security practitioners collect a new up to date dataset and LUT13 is one of them.

TABLE VI. COMPARISON BETWEEN LUT13 DATASET AND KDD CUP 99 DATASET

Algorithms	**Detection rate**	
	LUT13	***KDD Cup***
ANN	98,2%	81.5%
FCM	94.7%	79.6%

V. CONCLUSIONS AND FUTURE WORK

In this paper, we discussed some challenges in developing a real NIDS. Motivated from the difficulty in building an up to date dataset, we proposed a scheme which is able to capture real-time traffic from the network to produce LUT13 dataset. We practically demonstrated the efficiency of our dataset and the obsoleteness of KDD Cup dataset by applying ANN and FCM algorithms to these datasets. The NIDSs which were trained on LUT13 dataset detects intrusive actions with a higher accuracy than that were trained on KDD Cup.

There are a number of potential researches that are arisen from this work. First, we would like to test more machine learning techniques on our dataset and compared them with KDD Cup and other datasets. Second, we plan to apply parallel methods to design our Packet Sniffer to speed up the performance of the system. Last but not least, we want to carry out experiments to remove some redundant features and add some more features to enhance the accuracy of detection rate of the system.

ACKNOWLEDGMENT

The Department of Network Security at the Faculty of Information Technology, Le Quy Don Technical University provided the facilities for this research.

REFERENCES

[1] W. Lee, S. Stolfo, K. Mok, "A Data Mining Framework for Building Intrusion Detection Models", Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 120-132, 1999.

[2] W. Lee, S. Stolfo, "A framework for constructing features and models for intrusion detection systems", ACM Transactions on Information and System Security, vol. 3, no. 4, pp.227-261, 2000.

[3] Shafi, K., Abbass, H. A., and W. Zhu (2009). "A Methodology to Evaluate Supervised Learning Algorithms For Intrusion Detection". School of ITEE, UNSW@ADFA, TR-ALAR-200904009.

[4] Kamran Shafi, Hussein A. Abbass, "Evaluation of an Adaptive Genetic-Based Signature Extraction System for Network Intrusion Detection". Journal of Pattern Analysis and Applications, DOI 10.1007/s10044-011-0255-5, Print ISSN 1433-7541.

[5] R. P. Lippmann, and M. A. Zissman. "1998 DARPA/AFRL off-line intrusion detection evaluation", dataset available at http://www.ll.mit.edu/IST/ideval/data/data index.html, 1998.

[6] KDD Cup 1999 data (Computer network intrusion detection). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[7] Maheshkumar Sabhnani, and G¨ursel Serpen, "Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set". Intelligent Data Analysis, 8(4):403–415, 2004.

[8] Z. Muda, W. Yassin, M.N. Sulaiman, and N.I. Udzir, "Intrusion detection based on K-Means clustering and Naïve Bayes classification", 7th International Conference on Information Technology in Asia: Emerging Convergences and Singularity of Forms (CITA), 2011.

[9] Wei Wang; Gombault, S.; Guyet, T., "Towards Fast Detecting Intrusions:Using KeyAttributes of Network Traffic", Internet Monitoring and Protection, 2008. ICIMP '08. The Third International Conference on Digital Object Identifier: 10.1109/ICIMP.2008.13 Publication Year: 2008 , Page(s): 86 – 91.

[10] Yang Li, Bin-Xing Fang, You Chen, Li Guo, "A Lightweight Intrusion Detection Model Based onFeature Selection and Maximum Entropy", Model Communication Technology, 2006. ICCT 06. International Conference on Digital Object Identifier: 10.1109/ICCT.2006.341771 Publication Year: 2006 , Page(s): 1 – 4.

[11] Panda, Mrutyunjaya; Patra, Manas Ranjan, "Some Clustering Algorithms to Enhance the Performance of the Network Intrusion Detection System", Journal of Theoretical & Applied Information Technology;2008, Vol. 4 Issue 8, p710

[12] Komviriyavut, T.; Sangkatsanee, P.; Wattanapongsakorn, N. ; Charnsripinyo, C.; "Network intrusion detection and classification with Decision Tree and rule based approaches", Communications and Information Technology, 2009. ISCIT 2009. 9th International Symposium on Digital Object Identifier: 10.1109/ISCIT.2009.5341005 Publication Year: 2009 , Page(s): 1046 – 1050.

[13] Quang Anh Tran; Jiang, F.; Jiankun Hu; "A Real-Time NetFlow-based Intrusion Detection System with Improved BBNN and High-Frequency Field Programmable Gate Arrays", Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on Digital Object Identifier: 10.1109/TrustCom.2012.51 Publication Year: 2012 , Page(s): 201-208.

[14] Winpcap library. Available from, http://www.winpcap.org.

[15] Risso, F. ; Degioanni, L.; "An architecture for high performance network analysis" Computers and Communications, 2001. Proceedings. Sixth IEEE Symposium on Digital Object Identifier: 10.1109/ISCC.2001.935450 Publication Year: 2001 , Page(s): 686-693

[16] DARPA Intrusion Detection Evaluation. Available from, http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/docs/detections_1999.html.

[17] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors". Nature, 323:533–536, 1986.