Cyberattacks Detection and Analysis in a Network Log System Using XGBoost with ELK Stack

Chao-Tung Yang (ctyang@thu.edu.tw)
Tunghai University https://orcid.org/0000-0002-9579-4426
Yu-Wei Chan
Providence University
Jung-Chun Liu Liu
Tunghai University
Endah Kristiani
Tunghai University
Cing-Han Lai
Tunghai University

Research Article

Keywords: Cyber Security , Machine Learning , ELK Stack , XGBoost , NetFlow Log

Posted Date: September 3rd, 2021

DOI: https://doi.org/10.21203/rs.3.rs-838650/v1

Version of Record: A version of this preprint was published at Soft Computing on March 31st, 2022. See the published version at https://doi.org/10.1007/s00500-022-06954-8.

Noname manuscript No.
(will be inserted by the editor)

Cyberattacks Detection and Analysis in a Network Log System Using XGBoost with ELK Stack

Chao-Tung Yang · Yu-Wei Chan · Jung-Chun Liu · Endah Kristiani · Cing-Han Lai

Received: date / Accepted: date

Abstract The usage of artificial intelligence and machine learning methods on cyberattacks increasing significantly recently. For the defense method of cyberattacks, it is possible to detect and identify the attack event by observing the log data and analyzing whether it has abnormal behavior or not. This paper implemented the ELK Stack network log system (NetFlow Log) to visually analyze log data and present several network attack behavior characteristics for further analysis. Additionally, this system evaluated the extreme gradient enhancement (XGBoost), Recurrent Neural Network (RNN), and Deep Neural Network (DNN) model for machine learning methods. Keras was used as a deep learning framework for building a model to detect the attack event. From the experiments, it can be confirmed that the XGBoost model has an accuracy rate of 96.01% for potential threats. The full attack data set can achieve 96.26% accuracy, which is better than RNN and DNN models.

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 108-2221-E-029-010, 109-2221-E-029-020, and 110-2221-E-029-020-MY3.

Corresponding author: ctyang@thu.edu.tw

C-T Yang, J-C Liu, and C-H Lai
Department of Computer Science,
Tunghai University, Taichung City 407224, Taiwan, (R.O.C.)

C-T Yang
Research Center for Smart Sustainable Circular Economy,
Tunghai University, No. 1727, Sec.4, Taiwan Boulevard,
Taichung City 407224, Taiwan, (R.O.C.)
E-mail: ctyang@thu.edu.tw

Y-W Chan
College of Computing and Informatics, Providence University, Taichung City 43301, Taiwan (R.O.C.)

E. Kristiani
Department of Industrial Engineering and Enterprise Information,
Tunghai University, Taichung City 407224, Taiwan, (R.O.C.)
Department of Informatics, Krida Wacana Christian University, Jakarta 11470, Indonesia

Keywords Cyber Security · Machine Learning · ELK Stack · XGBoost · NetFlow Log

1 Introduction

In recent years, cyberattacks are evolving and becoming more sophisticated. For example, with the development of Machine Learning algorithms, some illegal users might use the technology of cyberattacks and Machine Learning to analyze information from the social networks [3]. The specific target of cyberattacks is given based on the data, the attack success rate, or the vulnerability that is discovered [30]. According to Neustar's International Network Benchmark Index report released in 2018 [1], 82% of cybersecurity experts said they are worried that attackers will use artificial intelligence to make a destructive attack on the network environment. However, a large number of experts believe that artificial intelligence can play a considerable role in network security and provides excellent supporting [26].

As mentioned above, in the campus network environment, various cyberattacks have appeared and tried to attempt the stability of the campus network environment. From the network logs, it can be found that many unusual network usage scenarios are trying to pass the campus network security system [15] [18] [28]. However, the systems with visualized network log data and the capability of detecting cyberattacks have considerable charges.

The open-source platform ELK Stack is implemented to build a network log system (NetFlow Log) in this work. First, the network logs related to the cyber attack behavior were collected in a large amount of data and then obtained preliminary information. Second, the data analysis was observed. After visualizing the log data, the administrator can use the machine learning model to import historical log data for analysis and detection. Then, perform a risk assessment based on the cross-validation analysis of the visual information displayed by the ELK Stack, even if it has not occurred or uncertain events. The administrator also has sufficient information to make the right decisions and take precautions to avoid the associated losses in information security. Our goal is to use XGBoost for machine learning, then implement a visualization system for cyberattack behavior to help administrators detect whether historical network log data has cyber attack behavior or not. The specific objectives are as follows:

1. Demonstrate the visualization and monitoring system of NetFlow log.
2. Compare XGBoost, RNN, and DNN model in two kinds of model, potential attack and full attack log data.

2 Background Review and Related Works

This section provides the background of this work and several kit information, including Python, ELK Stack, XGBoost, and so on. Then, the next section is discussed in more detail.

2.1 Keras

In Deep Learning, The Microsoft Cognitive Toolkit (CNTK) and TensorFlow are widely used in Deep Learning research. However, although both have compelling features, the actual application is more complicated. Therefore, the Deep Learning project for this job will use Keras to build a Deep Learning model.

Keras is an open-source neural network library written in Python that can be executed on TensorFlow, CNTK, Theano. The leading developer is Google engineer Francois Chollet. Keras can quickly implement deep neural networks.

2.2 ELK Stack

ELK Stack refers to the architecture based on three open-source software Elasticsearch, Logstash, and Kibana [6]. ELK Stack can be used to form a system for querying, collecting, and analyzing logs. This work can get data from any source and format. Without changing the original system architecture, ELK Sack is built to instantly search and analyze data and ultimately use visual capabilities to present the analyzed data results [21]. NetFlow Log is the automated network log platform built in this work. It is built on top of these three open-source software. In addition, ELK Stack has three kits and many other software packages, such as Filebeat, Xpack, and ECE.

2.3 Decision Tree

In decision theory, a decision tree consists of decision graphs and possible outcomes that are used to help decision-making achieve program goals [22]. In Machine Learning, a decision tree is a predictive model. Use tree graphics to help computers judge, segment our data, and make decisions based on it. Each node in the tree represents a specific target, and each forked path represents a possible feature of data segmentation. The information gain is obtained from the action of segmenting the data. The segmentation process is repeated until the leaf node. This leaf node corresponds to the target from the master node to the leaf node and has all the feature values on the path. Decision trees are available for forecasting and data analysis. A complete decision tree typically contains three nodes: decision nodes, opportunity nodes, and endpoints. Decision trees have several generation methods, classification tree analysis, regression tree analysis, Classification and Regression Trees (CART), Chi-square Automated Interaction Detection (CHAID).

As the most fundamental component of XGBoost, it needs to introduce the CART regression tree. It constructs a decision tree based on the characteristics and data of the training to determine the prediction result of each piece of data. Also, it uses the Gini index to calculate the gain to select the characteristics of the decision tree. The Gini index formula is as follows:

$$Gini(D) = \sum_{k=1}^{K} p_k 1 - p_k \tag{1}$$

p_k represents the probability of class classification category (k) in dataset D, the number of categories is indicated by K. The Gini index calculates the gain formula as:

$$Gini(D,A) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2) \tag{2}$$

D represents the entire dataset, D_1 and D_2 respectively represent data having feature A in the dataset and data other than A.

2.4 Gradient Boosting

Gradient Boosting is a Boosting method that is a Machine Learning technique for regression and classification problems [11]. Gradient Boosting generates prediction models in the form of multiple weak classifiers. Each model is established in the gradient direction of the loss function of the previous model. Put, when the loss function is large, the model is more error-prone. On the other hand, if our model can make the loss function drop, our model will continue to improve. Thus, the loss function is reduced in the gradient direction by multiple improvements, and a good model is finally obtained [12]. The specific algorithm is as follows:
Input Training set $T = (x_1, y_1), (x_2, y_2)..., (x_n, y_n)$
Output Boosting tree $f_M\ (x)$
Procedures:

- Initialization $f_0(x) = 0 form = 1, 2..., M$
- Calculating the residual $r_m i = y_i - f_(m-1))(x_i), i = 1, 2, ..., n$ (3)
- Fitting the residual $r_m i$ to learn a regression tree and get T(x:Θ_m)
- Update $f_m(x) = f_(m-1))(x) + T(x : \Theta_m)$
- Get the regression boosting tree
  $f_M(x) = \sum_{m=1}^{M} T(\frac{x}{\Theta_m})$

2.5 XGBoost

The objective function of XGBoost consists of two parts [8]. The first part is used to calculate the difference between the predicted score and the true score is $Obj(t) = \sum_{i=1}^{n} L(y_i, \hat{y}^{t-1} + f_t(x_i)) + \Omega(f_t) + constant$ The second part is normalization $\Omega(f_t)$, and the formula is as follows.
$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{i=1}^{T} w_j^2$

T represents the number of leaf nodes, w_j represents the weight of the j leaf nodes, γ control the number of leaf nodes, λ control the score of the leaf nodes not too large to prevent overfitting.

2.6 Deep Neural Network (DNN) and Reccurent Neural Network (RNN)

The generalized DNN contains variants such as CNN and RNN. In practical applications, the so-called Deep Neural Networks usually incorporates several known structures, such as LSTM or convolution layer. However, in a narrow sense, the difference between DNN and RNN and CNN is that DNN is especially expressed as a fully connected neuron structure and does not contain convolution units or temporal associations. DNN is sometimes called Multi-Layer perceptron (MLP)

The neural network is used to process sequence data is called RNN. In the neural network model of DNN, the neural layers are fully connected, but the nodes of each layer are not connected. This neural network model is very inefficient in processing sequence problems. For example, in advertising promotion, one needs to understand the user's browsing habits or preferences and use them. The principle of the RNN model is to connect the neuron's output back to the neuron's input. The network memorizes the previous message and uses it for the calculation of the current output. That is to say, the output of RNN is affected by the input of the last layer and the output neuron same layer.

2.7 Related Works

There are many theories, ideas, and experimental structures of other research, which allowed us to have better results in our experiments. According to the background of this work [19], Iman Sharafaldin et al. [24] gave us a lot of inspiration, also analyzed a large amount of data and visualized it, and proposed a classification of cyberattacks. In addition, at the IEEE International Conference on Smart Computing (SMARTCOMP) in 2017, a conference paper published by X. Yuan et al. mentioned [29] the defense mechanism of DDos and its use of Deep Learning to establish a DDoS attack, also given us inspired. In addition to these, many papers give us a lot of constructive references [9].

In a paper published by Rafał Kozik et al. [14], the flexibility of cloud-based architecture was used for large-scale Machine Learning, shifting high computing requirements and high-storage parts to the cloud. The cloud-first builds a complex learning model and then uses edge computing to execute it.

In a paper published by Muhammad Al-Qurishi et al. [4], a model for predicting Sybil attacks using Deep Learning is proposed. The Sybil attack denies the reception or transmission of real nodes on the network by creating enough false identities, effectively blocking the network services of other users. Through its experiments, it is possible to provide high-precision predictions even when importing uncleaned data effectively. The campus network security system offers the network log data used in this work. The training and prediction are raw data. Through experiments, even complete attack behavior data can be high accuracy without error judgment.

James Zhang et al. have proposed a method to detect abnormal behavior of network performance data [31], which uses Open Science Grid to collect and use perfSONAR servers and uses Boosted Decision Tree (BDT) and simple feedforward neural networks for Machine Learning. In this work, eXtreme Gradient Boosting is also

used for decision classification to detect anomalous behavior in network log data. The network log data is divided into attack and non-attack and finally submitted to ELK for visualization analysis.

Today's hackers can use HTTP Parameter Pollution [2] training data to achieve classification that undermines Machine Learning and input design data into training data to reduce detection accuracy. The paper published by Sen Chen et al. proposes a two-stage learning enhancement method KUAFUDET [7] to learn and identify malware through confrontation detection. It includes the training phase of selecting and extracting features and the testing phase of using the first training phase. The sorting extraction of feature importance was also used in their work, and the complete attack data and the general original log data were imported as experimental data for reference comparison.

Hongyu Liu et al. have proposed a point-to-point detection method [17]. Based on the Deep Learning model of convolutional neural networks and recurrent neural networks, payload classification (PL-RNN) is performed and used for attack detection. XGBoost is used in this work to learn log data and summarize its important features. It effectively detects the difference between normal data and aggressive behavior and serves as the basis for both classifications. In addition, a paper published by Peiyuan Sun et al., [25] a Machine Learning-based approach was proposed, which can model the attack behavior based on intuitive observation.

Ibrahim Ghafir et al. have proposed a Machine Learning-based system [13] that can detect and predict APT attacks accurately and quickly. The system can be evaluated experimentally, and APT can be predicted in an early step. The prediction accuracy rate is 84.8%. Machine Learning is also used to quickly build a predictive model to classify network logs in this work. It has half of the cyber attack behavior and has high accuracy. In addition, this work has constructed a visualization system that provides network log data so that administrators can easily view log data at any point in time.

The paper presented by Ozgur Koray Sahingoz et al. [23] mentions that phishing is one of the methods used by hackers today. It proposes a real-time anti-phishing system, which has been experimentally proven to detect the network. Authentic rate of 97.98% when phishing URL

The paper presented by Abebe Abeshu Diro and Naveen Chilamkurti [10] mentions that applying Deep Learning for attack detection is the preferred approach because of its high feature extraction capabilities. In their work, they also hope Machine Learning can make progress in detecting attacks.

At the 2015 International Conference on Information and Communication Technology and Systems (ICTS), P. P. I. Langi et al. presented an assessment of Logstah and Elasticsearch [16]. The managers of the Institute of Nuclear Physica, Italy (INFN), used ELK Stack to set up a monitoring system to facilitate the management of each node's activities [5].In a conference paper, T. Ram Prakash et al. proposed the construction of the ELK Stack system and how to identify network users [20] geographically. In addition, the paper by Chao-Tung Yang et al. [27] also proposed a visual platform system using ELK Stack as a statistical analysis of air quality and influenza-like illness. This work refers to the ELK Stack construction method and finally success-

fully imports the network log of the campus network security system and analyzes the data.

3 System Design and Implementation

This chapter describes how to use artificial intelligence to build predictive models and use ELK Stack to visualize system architecture and network log data implementation. In addition, this work creates a Deep Learning model using DNN and RNN to compare with the XGBoost Machine Learning model. The network logs collected in this work are based on campus network devices, with more than 7 million data per day, approximately 2 to 3G. 2 TB has been collected and continues to increase.

3.1 System Architecture

In this work, we installed Anaconda3 on Windows 10 and use Juypter Notebook as the Python development environment. After pre-processing the network log data in the development environment, using XGBoost for Machine Learning and execute historical network log data to check the cyber-attack behavior. In addition, construct a network log system on Linux systems using open source software such as ELK Stack to visualize the cyber attack behavior for more intuitive analysis by managers.

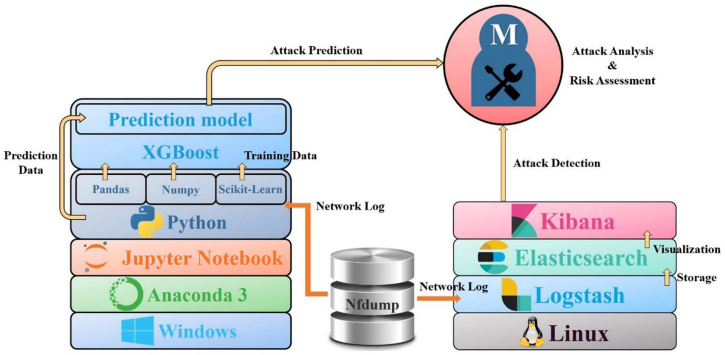

Fig. 1: System architecture

As shown in Figure 1, the network logs are collected and submitted to the ELK for visual analysis to present the results of the cyber attack behavior detection to the administrator. On the other hand, Python imports network logs, perform data preprocessing, and conducts model training. Finally, the model submits the cyber attack prediction result to managers. Suppose the ELK Stack analyzes the log data into a regular data stream. Still, the model prediction results show that the data stream is an attack behavior. In that case, the administrator can use the results of both parties for cross-validation analysis to perform the risk assessment. It can prevent the impact of hidden cyber attacks or unknown cyber-attacks.

3.2 NetFlow Log System

First, Linux built-in shell scripts were used to write scripts and schedules so that the machine can automatically download the network log data from the server-side. This server-side collected NetFlow log using Netdump. NetDump is a tool that catches all types of packets on our LAN network and prints them out. This tool aims to acquire information and categorized the different packets that flow on the LAN. After data processing, Logstash collects and filters the log data. Then Logstash is transferred to Elasticsearch for later data search or analysis. Then Kibana is used to visualize the analyzed data and finally present it on the website. The above is the NetFlow Log System, a campus network log platform.

3.2.1 Network Usage

Before analyzing the cyber attack behavior, this work can set up several frequently used domains and visualize the log data. The administrator can monitor the network for abnormal use. In addition, this paper divides these domains into search engines, auction sites, online communities, entertainment, and high-risk domains. All of the above domain IPs are public IPs and can only be observed by the administrator.

3.2.2 Attack Data Analysis

Cyber-attacks tend to hide their packaging and pretend to be a secure data stream to trick the information security system. However, just like walking in the snow, we will leave footprints. This work selected several kinds of cyber-attacks and recorded their eigenvalues. Then use Elasticsearch to filter the cyber log data. Managers monitor data visualization of suspected cyber attacks.

3.3 Machine Learning with XGBoost

This section discusses how to use XGBoost for Machine Learning and construct a prediction model to detect the cyber-attack behavior in network log data. Also, determine which data streams in the network log data are suspected of having cyber attacks behavior and which are normal. Figure 2 shows the decision tree of XGBoost.

3.4 Machine Learning with DNN

Figure 3 is the DNN model established by this work, including an input layer and the final output layer. It also contains two hidden layers and three dropout layers, which are fully connected.

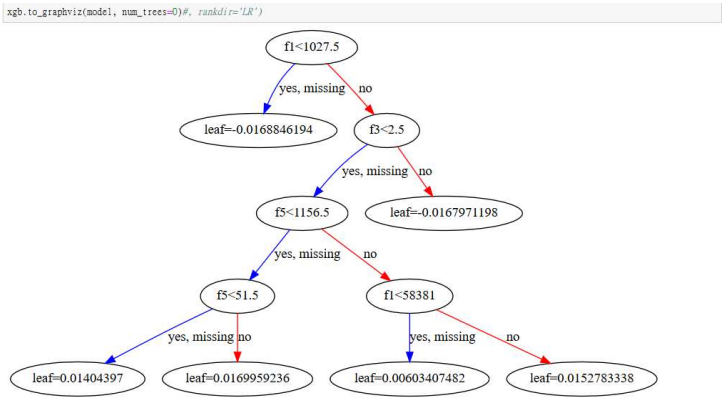

Fig. 2: XGBoost decision tree

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 128)	1024
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 256)	33024
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 128)	32896
dropout_3 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 1)	129

Total params: 67,073
Trainable params: 67,073
Non-trainable params: 0

Fig. 3: DNN model

3.5 Machine Learning with RNN

Figure 4 is the RNN model established by this work. The difference between the RNN model and the DNN model is that the output of the RNN is not only affected by the input of the previous layer, but also by the output of the same layer of neurons.

3.6 Data Preprocessing

First, the log data must be preprocessed to convert the data to a format that the machine can learn. The algorithm is as follows.1. In addition, our log data has raw data of 500,000 records, and the data of suspected aggression accounts for about 1.8%

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_1 (SimpleRNN)     (None, None, 128)         17408
_________________________________________________________________
dropout_1 (Dropout)          (None, None, 128)         0
_________________________________________________________________
simple_rnn_2 (SimpleRNN)     (None, None, 256)         98560
_________________________________________________________________
dropout_2 (Dropout)          (None, None, 256)         0
_________________________________________________________________
simple_rnn_3 (SimpleRNN)     (None, 128)               49280
_________________________________________________________________
dropout_3 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 165,377
Trainable params: 165,377
Non-trainable params: 0
_________________________________________________________________
```

Fig. 4: RNN model

of the total number of single log data. This experiment uses ELK Stack to filter the attack data of different periods and then extract the log data from the database for integration. Finally, the log data is pre-processed to complete the pre-operation of the training set and the verification set. A total of about 200,000 records is divided into 66% as a training set and 33% as a verification set. Therefore, there are two kinds of the dataset to be trained, raw and full attack log data.

Algorithm 1 Data Preprocess for Prediction model

Require: Nfdump.txt log data from the campus network environment, *Dataset*;
Ensure: *Training*, *Test*, and *Prediction* Dataset;
1: Python read *Dataset*;
2: Let column in *Dataset* = *data feature*;
3: Del *unneeded data* and *blanks*;
4: **if** (data = cyber attacks behavior) **then**
5: *mark data* = *Attack* = 1;
6: **else**(data != cyber attacks behavior)
7: *mark data* = *Normal* = 0;
8: **end if**
9: *return Dataset*;
10: *Training Dataset* = *Split* 66%*Dataset*;
11: *Test Dataset* = *Split* 33%*Dataset*;
12: *Prediction Dataset* = *full Dataset*;

3.6.1 XGBoost Model Training

Undertake the preprocessing data of 3.3.1, and then import the data into the model for Machine Learning training. However, compared to data with cyber-attacks, standard

network usage data accounts for most of the logs and may not even appear. Therefore, how to make the model learn the correct features is the primary goal.

The data of cyberattack behavior is classified as normal traffic or noise to avoid Machine Learning to classify data. Therefore, collecting log data for multiple periods and filter out the data with attack characteristics to form a training set. Our training set will try to train by writing data from different attacks and non-attacks. Finally, both attack and non-attack data contain approximately 50% of the data in this work, providing the best model feedback. The training set includes roughly 150,000 log data, and the validation set contains 50% of the data, including attack data and non-attack data, for a total of approximately 77,000 data. In addition, using random floating parameters to adjust the parameters in XGBoost, use L1 and L2 normalization to perform regular gradient enhancement, avoid overfitting or inappropriate. The feature importance is passed after each training to adjust the characteristics of the log data.

3.6.2 XGBoost Model Prediction

In the forecast set, use two types of data to import Machine Learning model predictions. The first is 96.26% complete attack data, and the second is new, unmodified log data. It verifies the correctness and versatility of our model. Finally, the training and validation of the model is completed, which will have high precision and a good F1 score. The algorithm is as follows (2).

Algorithm 2 The Prediction model for cyber attacks

Require: *New raw log Data* from $Nfdump.txt$;
Ensure: The *amount* of predicted data for cyber attacks;
1: Upload $Nfdump.txt$ to website;
2: Python read *New raw log Data*;
3: *New raw log Data* do Data Preprocess;
4: Load *Prediction model*;
5: **if** (*data* = cyber attacks behavior) **then**
6: $Count\ data = Attack$;
7: **else**(*data* != cyber attacks behavior)
8: $Count\ data = Normal$;
9: **end if**
10: *return Count*;
11: *validation accuracy*

3.7 Deep Learning with Keras

In this section, we discuss how to use Keras for Deep Learning, In the field of Deep Learning, CNTK and TensorFlow are widely used in Deep Learning research. However, although both have compelling features, the actual application is more complicated. Therefore, the Deep Learning project for this job will use Keras to build a dichotomy prediction model, perform the cyber attack behavior detection on network

log data, and determine which data streams in the network log data are suspected of having the cyber attack behavior and which are normal. In this work, the DNN model and the RNN model were built, and they will have experimented with the same data as the XGBoost Machine Learning model.

3.7.1 Deep Neural Networks Model

First, the log data is pre-processed as in Section 3.3.1, and the data is converted into a format that the machine can learn. After the data is imported into the DNN model, it is trained in a supervised learning manner. In order to ensure that the DNN model can produce a globally optimal solution during the experiment, this work uses the Scikit-learn suite to optimize the parameters in the model. Since this work aims to predict potential attacks, there are several types of attacks in the data set. However, the characteristics of cyberattack behavior are very scattered, leading to over-fitting or gradient disappearing even after numerous adjustments. The data set is also given a full attack record to ensure the fairness of the experiment, as well as a new, unmodified log data resource for validation.

3.7.2 Recurrent Neural Network Model

In addition to the DNN solution, the data problem in the Deep Learning model also has RNN. Since DNN cannot fully predict full attack data, and there is often over-fitting or gradient disappearance. RNN can also deal with data problems and significantly improve DNN over-fitting. Therefore, this work is connected to the DNN model to rebuild an RNN model and give the same data to conduct experiments. In order to ensure that the RNN model can produce a globally optimal solution during the experiment, this work uses the Scikit-learn suite to optimize the parameters in the model.

4 Experimental Results

This section describes the use of XGBoost to build a Machine Learning model, Keras to build DNN and RNN Deep Learning models for binary classification prediction, and ELK Stack to analyze network usage and attack behavior characteristics.

4.1 Experimental Environment

This section describes our hardware lab environment. This experiment uses two hosts, one with Linux as the operating system and the ELK Stack server. The other is to use Window10 as the operating system, install Anaconda 3 and related kits in the Python development environment, and build the XGBoost Machine Learning model. Detailed hardware devices are shown in Table 1.

Table 1: Hardware specifications

Item	Disk	Core	Ram	OS
NetFlow Log	8TB	10 CPUs x Intel(R)Core(TM) i7-6950X CPU @ 3.00GHz	128G	Ubuntu 18.04
Machine Learning	1TB	Intel(R)Core(TM) i7-7700 CPU @ 3.60GHz	16G	Windows 10

4.2 ELK Stack Network Usage

To more easily confirm the network usage on campus, this experiment finds the public IP addresses of major commercial websites, search engines, social networks, etc., through the Internet. This information can be easily found on websites such as ipinfo.io. Then, use Elasticsearch to filter the required domain information, remove the non-service local IP address to avoid information miscellaneous, record the necessary domain name, and use Kibana to visualize it. Figure 5 shows the pie chart of Network usage.

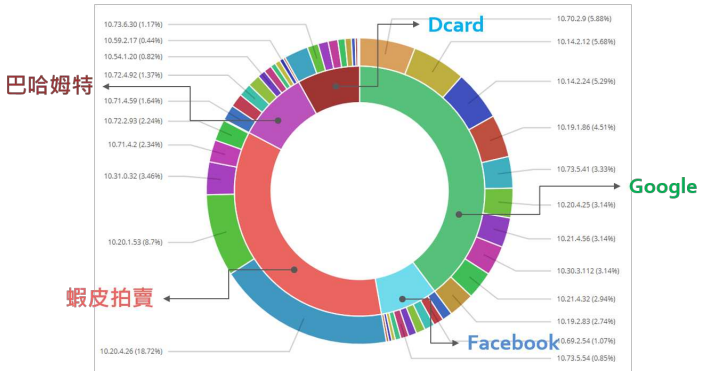

Fig. 5: Network usage

4.3 ELK Stack Attack Analysis

In this experiment, the characteristics of several kinds of cyber attack behaviors are selected as the screening conditions. After ELK analysis, the data is visualized and presented, providing an intuitive way for the administrator to observe the cyber attack. As Figure 6, Figure 7, Figure 8, and Figure 9 shown below.

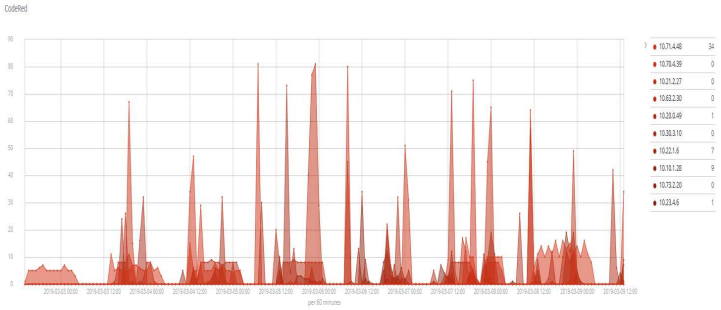

Fig. 6: The Graph of CodeRed attack

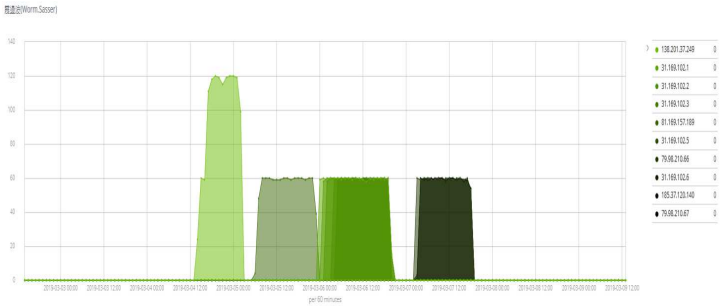

Fig. 7: The Graph of Worm Sasser attack

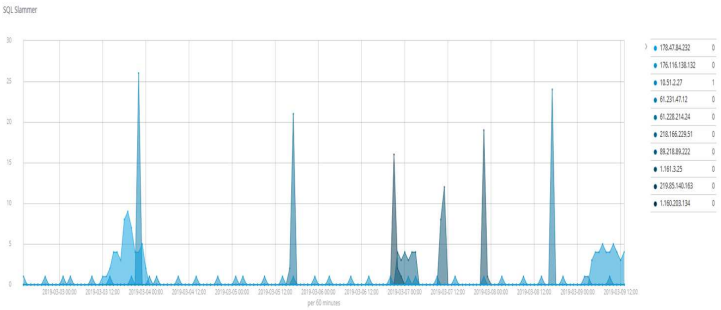

Fig. 8: The Graph of SQL Slammer attack

4.4 Machine Learning Data Preprocessing

There are about 500,000 data per data in the network log data, and the data of suspected aggression accounts for about 1.8% of the total. The experiment will have the best training results after about 50% of each experimental attack and non-attack to achieve better training conditions. Therefore, this experiment uses ELK Stack to filter other periods' attack data and then extract the log data from the database for

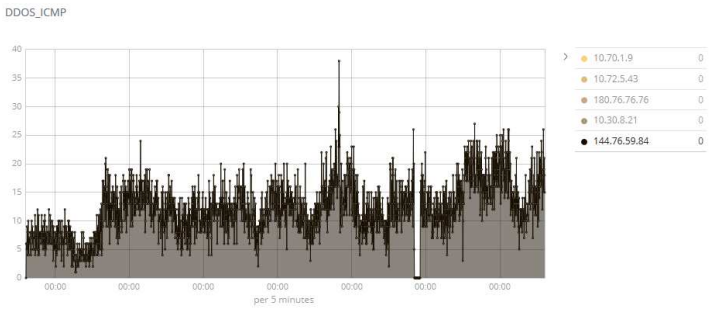

Fig. 9: ICMP DDOS

integration. Finally, the log data is pre-processed to complete the pre-operation of the training set and the verification set. About 200,000 pieces of data will be divided into 66% as a training set and 33% as a verification set.

4.5 XGBoost Model Prediction

Fig. 10 is a bar graph in which the feature importance is sorted according to the score. In order ”Dst Pt”,”In Byte”,”Src Pt”,”Output”,”In Pkt ”,”Duration”,”Proto”,”Input”.

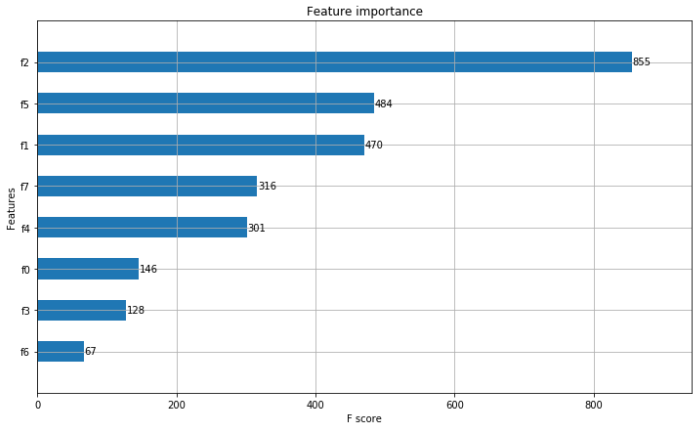

Fig. 10: Graph of feature score

As shown in Figure 11, Gain represents the relative contribution of the feature to the model, and a high value means that it is more important for prediction.

Weight indicates the number of times the feature is used to split the node. As shown in Figure 12.

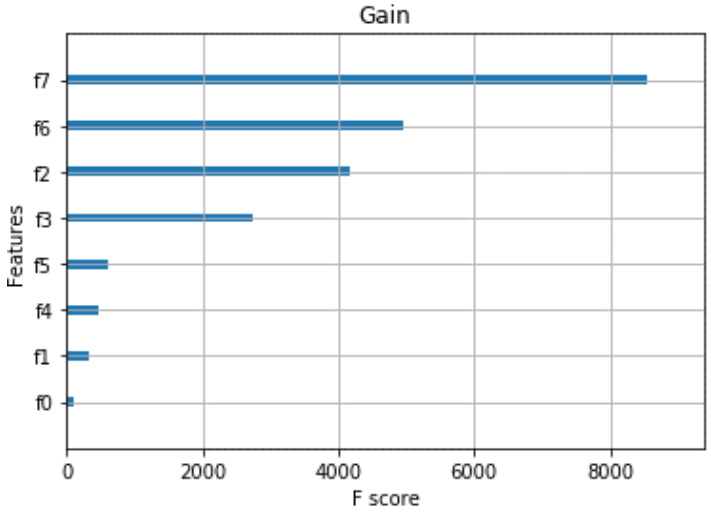

Fig. 11: Gain pyplot

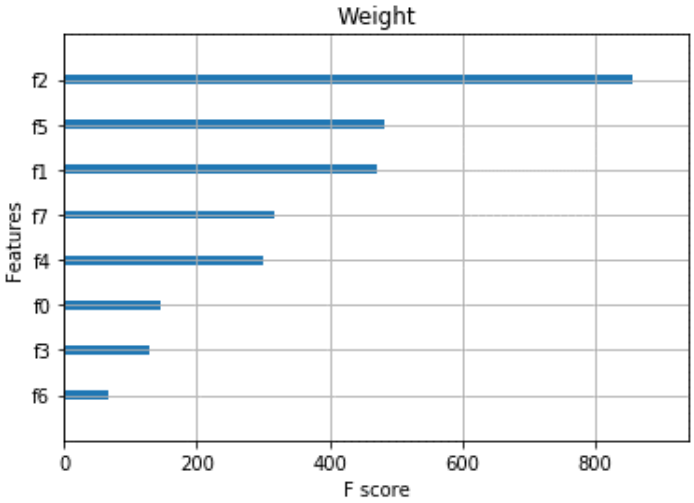

Fig. 12: Weight pyplot

Figure 13 represents the relative number of observations associated with this feature, for example, 100 observations, 4 features, and 3 trees, assuming f1 is used to determine 10, 5, and 2 observations in t1, t2, and t3, respectively. Calculate the coverage of this feature as 10 + 5 + 2 = 17 observations.

Total Gain represents the total gain that a feature brings in each split node in all trees as shown in Fig. 14. The number of all samples covered by a feature at each split node is called Total Cover as shown in Figure 15.

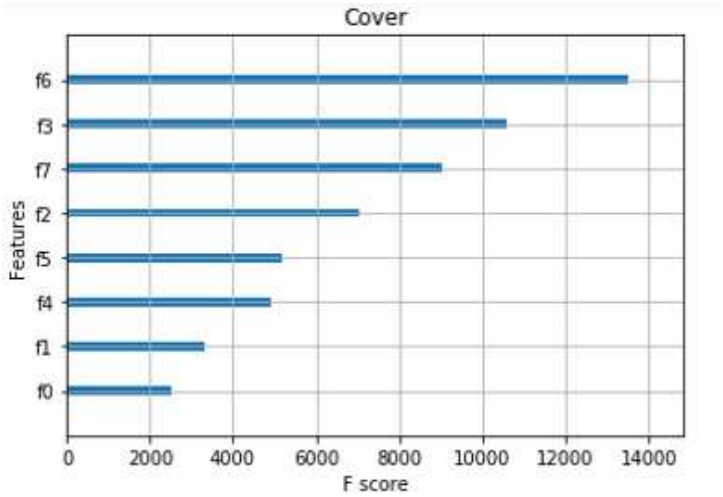

Fig. 13: Cover pyplot

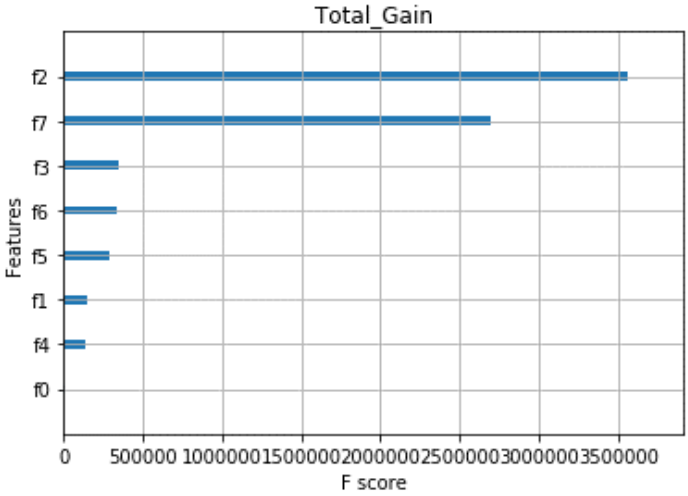

Fig. 14: Total Gain pyplot

To verify the correctness and versatility of the model, the data used in the prediction is the new raw log data, and the cleaned data is handed over to the model after preprocessing. The predicted result is as high as 96.01%. To verify the correctness of the model, a set of full-attack prediction sets is re-sampled here, and the accuracy rate is as high as 96.26%. It proves that the attack data can be fully recognized when attack behavior characteristics are in the log data. As shown in Tabel2.

Finally, the evaluation indicator is applied to test the model's mean square error (MSE), model accuracy, and F1 Score model correctness, as shown in Tabel 3.

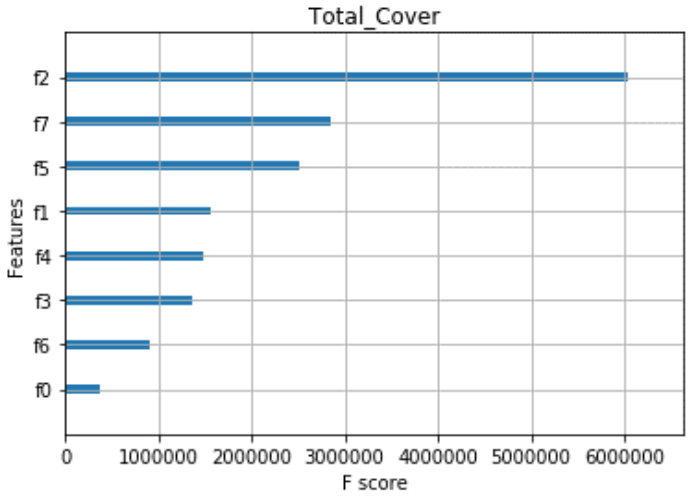

Fig. 15: Total Cover pyplot

Table 2: Model predictions

Predictions Dataset	Predictions Accuracy	Predictions Count		
New raw log data	96.01%	Item	Attack	Normal
		Val label	6,342	127,424
		Model Preds	11,679	122,087
Full attack log data	96.26%	Item	Attack	Normal
		Val label	5,679	0
		Model Preds	5,679	0

Table 3: Model score

Evaluation index	Score
MSE	2.53%
Accuracy	97.47%
F1 Score	97.54%

4.6 DNN Model Prediction

Figure 16 shows the training and validation loss values for this DNN model. It can be seen from the figure that the loss value of the training data keeps decreasing and is infinitely close to the validation data.

Figure 17 shows the training and validation accuracy values for this DNN model. From the figure that the accuracy of the training set is constantly increasing and close to the verification set. This is a good model.

The model validation set prediction results are shown in Tabel 4. The data used for prediction is the same as the data used by XGBoost. The DNN model predicts results as high as 96.89%. In order to verify the versatility of the model, a set of

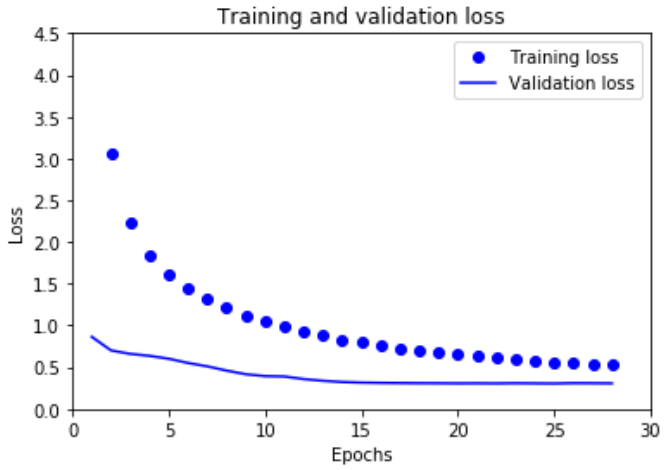

Fig. 16: DNN training and validation loss

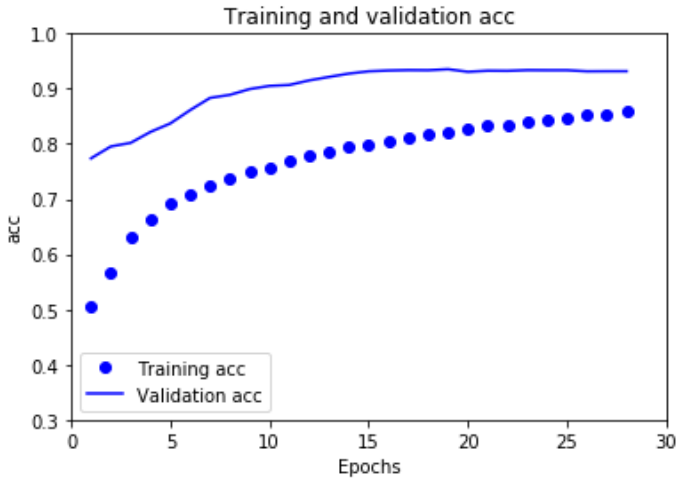

Fig. 17: DNN training and validation accuracy

full attack prediction sets is also sampled here, with an accuracy of only 69.66%. Compared with the previous accuracy record, it is very unexpected for such a result. The experimental result of this DNN model has the best result.

4.7 RNN Model Prediction

Figure 18 shows the training and validation loss values for this RNN model.Figure 19 shows the training and validation accuracy values for this RNN model. It can be

Table 4: DNN model predictions

Predictions Dataset	Predictions Accuracy	Predictions Count		
Y_test data	93.18%	**Item**	**Attack**	**Normal**
		Val label	**45587**	**45610**
		Model Preds	**45711**	**45486**
New raw log data	96.89%	**Item**	**Attack**	**Normal**
		Val label	**567122**	**4373**
		Model Preds	**558100**	**13395**
Full attack log data	69.66%	**Item**	**Attack**	**Normal**
		Val label	**5679**	**0**
		Model Preds	**3956**	**1723**

seen from these two figures that this RNN model is also training in a good direction, and has a good accuracy rate.

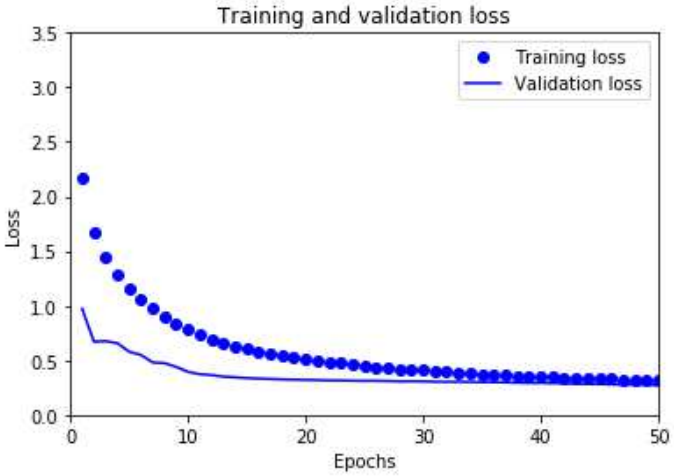

Fig. 18: RNN training and validation loss

The prediction results of the RNN model are shown in Tabel 5. The data used for prediction is the same as the data used in the first two models. The RNN model predicts results as high as 97.61%, even surpassing the accuracy of XGBoost. In order to verify the versatility of the model, the same set of attack data were also used for prediction, however the accuracy was only 70.85%.

The comparison of three models of XGBoost, RNN, and DNN is presented in the table 6.

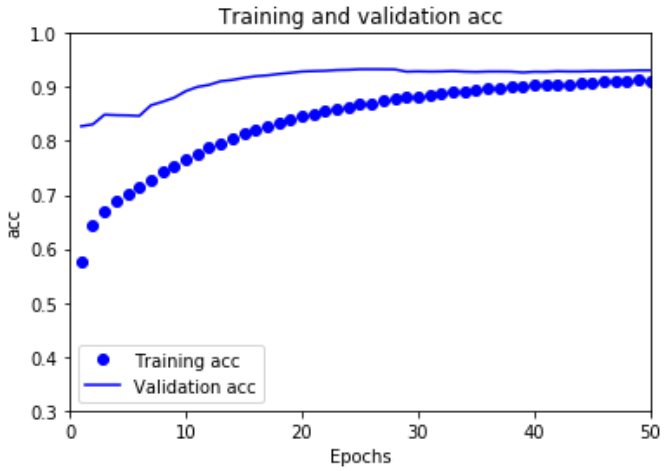

Fig. 19: RNN training and validation accuracy

Table 5: RNN model predictions

Predictions Dataset	Predictions Accuracy	Predictions Count		
Y_test data	93.34%	**Item**	**Attack**	**Normal**
		Val label	**45587**	**45610**
		Model Preds	**45786**	**45411**
New raw log data	97.61%	**Item**	**Attack**	**Normal**
		Val label	**567122**	**4373**
		Model Preds	**562197**	**9298**
Full attack log data	70.85%	**Item**	**Attack**	**Normal**
		Val label	**5679**	**0**
		Model Preds	**4024**	**1655**

Table 6: Comparison of experimental results of three models

Predictions Accuracy			
Model	**XGBoost**	**DNN**	**RNN**
New raw log data	96.01%	96.89%	97.61%
Full attack log data	96.26%	69.66%	70.85%

5 Conclusions and Future Works

This paper demonstrates a network log system monitoring and visualization using ELK Stack. This system allows administrators to easily visualize the charts and monitor the information they need from tens of millions of log data. This work also compares Machine Learning with Deep Learning models of XGBoost, RNN, and DNN. From the experimental results, XGBoost is the best in the data prediction of the full

attack. Therefore, this work chooses to use XGBoost as the machine learning model for the log data attack prediction. This attack prediction model can help to detect the ELK as the analyzed data. For example, suppose the ELK Stack analyzes a log as ordinary data. The model prediction results show that these data streams have aggressive behavior characteristics. In this case, the administrator can use the two-party results to cross verification and further information security risk assessment.

In the future, ELK Stack will collect more functional values related to the attack behavior and further visualize the Network log data as an analysis chart. Network usage will add the remaining large domain IP domains to it and distinguish each different domain. Convenient for management to observe. XGBoost is one of the most popular machine learning models. Its limitations are not limited to the two categories of attack and non-attack log data. It can more actively increase the data characteristics of the attack behavior, enrich our database. Use XGBoost to create a multi-classification model that can directly identify the type of attack and find unusual data from the network log. Besides, cross-validation can be used in conjunction with deep learning to compare predictions and improve information security.

Acknowledgments

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 108-2221-E-029-010, 109-2221-E-029-020, and 110-2221-E-029-020-MY3. The part of data has been presented previously in a conference proceeding at: https://doi.org/10.1007/978-981-15-3250-4_36.

Conflict of Interest

The authors declare that there is no conflict of interest.

References

1. Eighty two percent of security professionals fear artificial intelligence attacks against their organization (2018). https://www.home.neustar/about-us/news-room/press-releases/2018/NISCOctober
2. How to detect http parameter pollution attacks (2021). Https://www.acunetix.com/blog/whitepaper-http-parameter-pollution/
3. Ahad, N., Qadir, J., Ahsan, N.: Neural networks in wireless networks: Techniques, applications and guidelines. Journal of network and computer applications **68**, 1–27 (2016)
4. Al-Qurishi, M., Alrubaian, M., Rahman, S.M.M., Alamri, A., Hassan, M.M.: A prediction system of sybil attack in social network using deep-regression model. Future Generation Computer Systems **87**, 743 – 753 (2018). DOI https://doi.org/10.1016/j.future.2017.08.030. URL http://www.sciencedirect.com/science/article/pii/S0167739X17300821
5. Bagnasco, S., Berzano, D., Guarise, A., Lusso, S., Masera, M., Vallero, S.: Monitoring of IaaS and scientific applications on the cloud using the elasticsearch ecosystem. Journal of Physics: Conference Series **608**, 012016 (2015). DOI 10.1088/1742-6596/608/1/012016
6. Bajer, M.: Building an iot data hub with elasticsearch, logstash and kibana. In: 2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), pp. 63–68. IEEE (2017)

7. Chen, S., Xue, M., Fan, L., Hao, S., Xu, L., Zhu, H.: Hardening malware detection systems against cyber maneuvers: An adversarial machine learning approach. CoRR **abs/1706.04146** (2017). URL http://arxiv.org/abs/1706.04146
8. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 785–794. ACM, New York, NY, USA (2016). DOI 10.1145/2939672.2939785. URL http://doi.acm.org/10.1145/2939672.2939785
9. Chen, Z., Jiang, F., Cheng, Y., Gu, X., Liu, W., Peng, J.: Xgboost classifier for ddos attack detection and analysis in sdn-based cloud. In: 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), pp. 251–256. IEEE (2018)
10. Diro, A.A., Chilamkurti, N.: Distributed attack detection scheme using deep learning approach for internet of things. Future Generation Computer Systems **82**, 761 – 768 (2018). DOI https://doi.org/10.1016/j.future.2017.08.043. URL http://www.sciencedirect.com/science/article/pii/S0167739X17308488
11. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Annals of statistics pp. 1189–1232 (2001)
12. Friedman, J.H.: Stochastic gradient boosting. Computational statistics & data analysis **38**(4), 367–378 (2002)
13. Ghafir, I., Hammoudeh, M., Prenosil, V., Han, L., Hegarty, R., Rabie, K., Aparicio-Navarro, F.J.: Detection of advanced persistent threat using machine-learning correlation analysis. Future Generation Computer Systems **89**, 349 – 359 (2018). DOI https://doi.org/10.1016/j.future.2018.06.055. URL http://www.sciencedirect.com/science/article/pii/S0167739X18307532
14. Kozik, R., Choraś, M., Ficco, M., Palmieri, F.: A scalable distributed machine learning approach for attack detection in edge computing environments. Journal of Parallel and Distributed Computing **119**, 18 – 26 (2018). DOI https://doi.org/10.1016/j.jpdc.2018.03.006. URL http://www.sciencedirect.com/science/article/pii/S0743731518302004
15. Lai, C.H., Yang, C.T., Kristiani, E., Liu, J.C., Chan, Y.W.: Using xgboost for cyberattack detection and analysis in a network log system with elk stack. In: International Conference on Frontier Computing, pp. 302–311. Springer (2019)
16. Langi, P.P.I., , Najib, W., Aji, T.B.: An evaluation of twitter river and logstash performances as elasticsearch inputs for social media analysis of twitter. In: 2015 International Conference on Information Communication Technology and Systems (ICTS), pp. 181–186 (2015). DOI 10.1109/ICTS.2015.7379895
17. Liu, H., Lang, B., Liu, M., Yan, H.: Cnn and rnn based payload classification methods for attack detection. Knowledge-Based Systems **163**, 332 – 341 (2019). DOI https://doi.org/10.1016/j.knosys.2018.08.036. URL http://www.sciencedirect.com/science/article/pii/S0950705118304325
18. Liu, J.C., Yang, C.T., Chan, Y.W., Kristiani, E., Jiang, W.J.: Cyberattack detection model using deep learning in a network log system with data visualization. The Journal of Supercomputing pp. 1–20 (2021)
19. Peterson, P.: Unmasking deceptive attacks with machine learning. Computer Fraud and Security **2018**(11), 15 – 17 (2018). DOI https://doi.org/10.1016/S1361-3723(18)30110-6. URL http://www.sciencedirect.com/science/article/pii/S1361372318301106
20. Prakash, T.R., Kakkar, M., Patel, K.: Geo-identification of web users through logs using elk stack. 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence) pp. 606–610 (2016)
21. Rattan, A., Kaur, N., Bhushan, S.: Standardization of intelligent information of specific attack trends. In: Progress in Advanced Computing and Intelligent Engineering, pp. 75–86. Springer (2019)
22. Safavian, S.R., Landgrebe, D.: A survey of decision tree classifier methodology. IEEE transactions on systems, man, and cybernetics **21**(3), 660–674 (1991)
23. Sahingoz, O.K., Buber, E., Demir, O., Diri, B.: Machine learning based phishing detection from urls. Expert Systems with Applications **117**, 345 – 357 (2019). DOI https://doi.org/10.1016/j.eswa.2018.09.029. URL http://www.sciencedirect.com/science/article/pii/S0957417418306067
24. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: An evaluation framework for network security visualizations. Computers and Security **84**, 70 – 92 (2019). DOI https://doi.org/10.1016/j.cose.2019.03.005. URL http://www.sciencedirect.com/science/article/pii/S0167404818308952

25. Sun, P., Li, J., Bhuiyan, M.Z.A., Wang, L., Li, B.: Modeling and clustering attacker activities in iot through machine learning techniques. Information Sciences **479**, 456 – 471 (2019). DOI https://doi.org/10.1016/j.ins.2018.04.065. URL http://www.sciencedirect.com/science/article/pii/S0020025518303311
26. Yang, C., Shi, Z., Zhang, H., Wu, J., Shi, X.: Multiple attacks detection in cyber-physical systems using random finite set theory. IEEE transactions on cybernetics (2019)
27. Yang, C.T., Kristiani, E., Wang, Y.T., Min, G., Lai, C.H., Jiang, W.J.: On construction of a network log management system using elk stack with ceph. The Journal of Supercomputing **76**(8), 6344–6360 (2020)
28. Yang, C.T., Liu, J.C., Kristiani, E., Liu, M.L., You, I., Pau, G.: Netflow monitoring and cyberattack detection using deep learning with ceph. IEEE Access **8**, 7842–7850 (2020)
29. Yuan, X., Li, C., Li, X.: Deepdefense: Identifying ddos attack via deep learning. In: 2017 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 1–8 (2017). DOI 10.1109/SMARTCOMP.2017.7946998
30. Zhang, D., Liu, L., Feng, G.: Consensus of heterogeneous linear multiagent systems subject to aperiodic sampled-data and dos attack. IEEE transactions on cybernetics **49**(4), 1501–1511 (2018)
31. Zhang, J., Gardner, R., Vukotic, I.: Anomaly detection in wide area network meshes using two machine learning algorithms. Future Generation Computer Systems **93**, 418 – 426 (2019). DOI https://doi.org/10.1016/j.future.2018.07.023. URL http://www.sciencedirect.com/science/article/pii/S0167739X18302267