International Journal of Information Security
https://doi.org/10.1007/s10207-021-00555-6

REGULAR CONTRIBUTION

An SSH predictive model using machine learning with web proxy session logs

Junwon Lee[1] · **Heejo Lee**[2]

Abstract
An adversary can use SSH communication as a route for information leakage or hacking. Many studies have focused on TCP header analysis to detect encrypted communication. However, SSH detection using TCP header analysis is limited when changing TCP port information or modifying components of the SSH protocol. Various machine-learning (ML) techniques have been introduced to enhance network traffic classification by analyzing TCP headers. Most ML-based traffic classification research has analyzed network packet flows. However, because of the complex structures and the various implementations of the TCP protocol, a lot of time and resources are required for the recombination of network packet flows. This paper presents a novel contribution to overcome the problems of network packet analysis that employs web proxy session logs, which do not require the recombination of packets to prepare a dataset for analysis. Moreover, we propose a hybrid predictive model that is useful for web proxy session log analysis. In the modeling process, we collected the web proxy logs from an actual network of ICT companies (more than 10,000 employees, Seoul, South Korea) and used the random forest and decision tree algorithms for the supervised learning. The detection rate (DR) for the training dataset was 99.9%, which is similar to or higher than that of other studies using ML and deep learning. Using the dataset of DARPA99, we proved that the DR and FPR for our proposed model were better than those achieved by Alshammari et al.'s model. We expect that the proposed predictive model can be used to block illegal attempts at SSH communication over HTTP CONNECT by changing the destination port and to detect novel illegal communication protocols.

Keywords Web proxy · SSH · HTTP CONNECT · TCP tunneling · Machine learning · Random forest · Decision tree · PCA

1 Introduction

Web proxies have been widely adopted in enterprise network environments to block hacking or otherwise harmful sites and prevent unauthorized file transfers to the external internet. Web proxy logs also provide useful information for analyzing anomalous traffic based on application session information contrary to the firewall [16]. Most web proxies can provide the functionality of a relay server and analyze and control HTTP methods (GET, PUT, POST, and PATCH). When a user connects to an HTTPS server that uses an SSL certificate, the web proxy can participate in the cryptographic communication process by delivering its SSL certificate to the user instead of the public certificate of the site.

The web proxy can prevent information leakage via HTTPS communication by analyzing the encrypted HTTP header. HTTP headers contain various features (e.g., method, URI, referrer) required to communicate with the server and be used to improve the detection performance of prediction models created by the network administrator for detecting C&C access and information leakage [9]. Furthermore, because web proxy logs can be quickly collected without requiring recombination in the actual network environment, abnormal actions can be incorporated into the detection model. However, some abnormal communications in the traffic through the web proxy do not use the general HTTP method. Generally, if a web proxy encounters a communication that it cannot analyze, it will send the traffic to the internet without further protocol analysis to ensure the avail-

✉ Junwon Lee
junimirang@korea.ac.kr

Heejo Lee
heejo@korea.ac.kr

[1] Department of Computer and Radio Communications Engineering, Korea University, Seoul, Korea

[2] Department of Computer Science and Engineering, Korea University, Seoul, Korea

Published online: 11 June 2021

ability of internet services. Using this functional vulnerability of web proxy systems, internal adversaries or hackers attempt to communicate with the outside using non-HTTP methods. To establish a TCP tunnel between the server and client via a web proxy, the client utilizes an HTTP CONNECT communication with the web proxy, because the web proxy does not control communications over HTTP CONNECT such as SSH and SSL VPN. Therefore, it is common to develop blacklists and whitelists based on the manufacturer's categorical information and to allow other forms of communication that are not found on either list. However, if a connection to a zero-day site not registered on the blacklist of the web proxy is attempted, the connection will not be blocked.

Attackers can use this vulnerability to exploit targets within networks that employ web proxies. In the worst-case scenario, attackers can use this route to successfully communicate from a malware-infected PC to a C&C server. Of the atypical communications over HTTP CONNECT, there are often a large number of SSH communications. SSH is a useful communication protocol used to transfer files or for remote access. Even though the firewall blocks SSH communications from the outside, tunnel communications established internally are not blocked, and a reverse remote connection is possible through the same tunnel. This type of tunnel is also a source of security threats from hackers. In addition, a malicious user inside the network can use the communication route to transmit information to an external server. The firewall can forward the session to TCP port 22 (widely used as an SSH service port) via the web proxy to block SSH sites not registered on the blacklist. Even though the destination TCP port 22 is blocked, adversaries can spoof the server's TCP port. Therefore, controlling SSH communication by blocking known TCP ports is not sufficient, and network traffic classification that does not consider TCP port information is required.

For known cryptographic protocols, web proxies can exchange certificates as the middle network, and data loss prevention (DLP) using deep packet inspection (DPI) can analyze atypical traffic and monitor the content sent to the outside. Because the DLP system closely inspects network flows, it has risks such as the excessive collection of sensitive information, such as personal data and related privacy breaches. Privacy policies such as GDPR [1] are thus a limitation when monitoring encrypted content. Article 5 of GDPR states that personal data should only be collected for specific purposes [19], while Article 25 dictates that the only personal data necessary for each specific purpose can be processed [12]. Data collection for network traffic analysis must thus be minimized to comply with these strict privacy policies.

Rather than collecting all data, including private content, it can be more effective to analyze the network flow characteristics. It is impossible to classify TCP tunnel communications such as SSH using the intuitive analysis of TCP headers and the payload of packets. In recent years, machine learning (ML) has emerged as an effective solution to traffic classification. However, ML classification using network packet flows has some problems, such as packet dropping, out-of-order delivery, and packet corruption in data transmission [22]. In addition, the computational resources required to collect packets, reassemble them, and take only the TCP/IP header in order to minimize the volume of collected data are high [24]. Because significant time and resources are needed to prepare a training dataset, it is difficult to quickly incorporate a dataset generated from the existing network into the predictive model for that network.

This paper thus introduces an ML-based SSH predictive model that detects SSH traffic used for information leakage and hacking without additional content analysis. We describe a novel approach that differs from previous research using network packet flows in the following ways:

- We utilize web proxy session logs to overcome difficulties in packet flow analysis and use a traffic dataset generated from the existing network.
- To compensate for insufficient features, we add new features that reflect the network environment and consider the association between these features.
- To increase the detection rate (DR) and lower the false-positive rate (FPR), we construct a hybrid predictive model that combines a random forest model with a decision tree model.

Even though the proposed predictive model uses web proxy logs, which are significantly smaller than network packet flows, the detection accuracy is higher than reported by previous studies. The proposed model analyzed 12.4 MB of web proxy logs as a substitute for 3.4 GB of network packet flows. When using the web proxy session logs, our model had a detection accuracy for the test dataset not used for training of 70.5% for the DR and 5.2% for the FPR. In the evaluation using DARPA 99 test dataset, the accuracy of 100% for the DR and 1.1% for the FPR, which was better than achieved in the Alshammari et al.'s study (98.3% DR and 1.7% FPR), respectively [4]. We collected the web proxy logs and preprocessed the text data to create a dataset of numerical values. We then developed a predictive model that detects SSH communications by analyzing the *browse time*, *traffic transaction ratio*, *transmission speed*, and *average session counts*. The random forest and decision tree prediction model were applied as the ML algorithm for the classification process.

This paper is organized as follows: Section 2 describes the difficulties of preparing a network packet flow dataset and the limitations in analyzing tunnel communications over HTTP CONNECT in a web proxy. Section 3 introduces the analytical techniques used to classify network traffic. Section

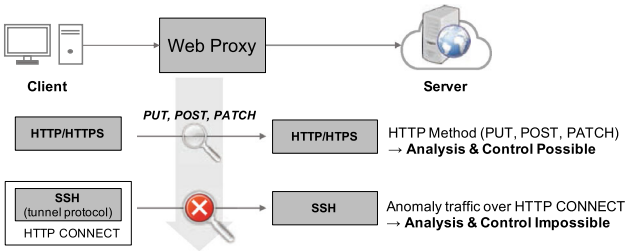


Fig. 1 SSH using TCP tunneling over HTTP CONNECT

4 proposes how to preprocess web proxy logs to develop an effective ML algorithm, and the ML-based SSH predictive model is introduced. Section 5 evaluates the DR and FPR using datasets not employed in the training process. Lastly, we conclude this paper with a discussion.

2 Problem analysis

2.1 Network packet flow datasets

Recent research on network analysis has been conducted based on network packet flow datasets. However, raw packet data processing is computationally intensive, and it takes too long to process large pcap files [24]. Therefore, the handling of large traffic volumes is a major challenge in the detection of malicious traffic [17]. In the present paper, the seven-day dataset collected to construct a detection model was 16 TB in size. Even if the Hadoop system was employed with the latest algorithm, it would take 195 h to reassemble the collected packets [27]. We also used the DARPA99 dataset of 4.87 GB to compare the network packet flows with other studies. In converting the network packet flows to a web proxy log format, it takes about 210 h to extract only the TCP/IP header information and convert it to the session dataset without recombining packets (Pentium 3.5 GHz; CPU: avg 40–50%; MEM: avg 9.76 GB). If the process of assembling packets is minimized, we can design an improved ML detection model. The present paper thus employs web proxy session logs, which are similar to TCP/IP headers but provide more information.

2.2 Encrypted SSH communication

As shown in Fig. 1, a web proxy analyzes and controls the HTTP methods. In general, web proxies can enhance the security of an internal network by providing HTTP header information for security threat analysis (Table 1).

However, the information in Table 1 has a null value shown in the communication over HTTP CONNECT. Encrypted tunnel communications over HTTP CONNECT between a server and a client cannot be analyzed with a web proxy, even though web proxies support SSL/TLS decryption. Therefore, tunnel communications over HTTP CONNECT need to be inferred and classified by analyzing the characteristics of the web proxy logs. However, in encrypted communications, the range of log features provided by a web proxy is relatively low compared to the features provided by network packet flows. Table 2 presents the features provided by a web proxy for HTTP CONNECT.

Table 1 Web proxy log fields for HTTP/HTTPS

Log column	Description
cs-bodylength	Number of bytes in the body(exclude header) sent from client to appliance
c-uri-pathquery	Path and query of the original URL request
x-cs(Referer)-uri	The URL from the Referer head
x-cs(Referer) -uri-query	Query from the 'Referer' URL

HTTP header analysis information is useful for the detection of security threats. However, if the HTTP method is not general, the information is not provided

3 Related work

Network traffic classification has been used to ensure the stability of stable network services and to prevent security threats. Recently, various techniques such as payload inspection and ML have been developed for this task as computational power increases.

3.1 Traffic classification using the port number

Because IP-Port is dedicated to one application, network traffic can be classified by recognizing a packet's header attributes [28]. In the TCP/UDP header, port numbers are provided based on the service to connect to the application. In general, applications use well-known ports defined by IANA, so it is possible to classify traffic by monitoring the destination port of a packet. Classification based on the port number analyzes the TCP header of packets and quickly classifies the services. Therefore, it is used to apply rules for access control and security policy in the switch or firewall. However, the server administrator can change the TCP/UDP port number, so an application can be disguised as another service by changing the port number for malicious purposes. There have been cases where TCP 80 and 443 ports were disguised as web services and used for RDP, SSH, TELNET, and FTP connections to bypass security control. Moreover, advanced evasion techniques (AETs) can disguise malicious payloads by sending frames across rarely used protocols [18] or violate common protocol forms to hide the actions of the attacker from firewall [10].

Table 2 Log fields from a web proxy for HTTP CONNECT

Log column	Description
date_time	GMT Date and time
browse_time	Calculated at run time from user session and stored as database field
c_ip	Client IP address
sc_bytes	Number of bytes sent from server to client
cs_bytes	Number of bytes sent from client to appliance
cs_host	Host name from the client's request URL
cs_uri_port	Port from the 'log' URL
cs_user_agent	Request header: User-Agent
sc_category	Category to which the site belongs
r_ip	IP address from the outbound server URL

Web proxy logs are similar to firewall logs. Because HTTP header analysis is possible, URL, user-agent, and site category information are available

3.2 Deep packet inspection

Because it is easy to change the TCP port number, payload analysis is required for accurate traffic classification. DPI analyzes plain text or encoded strings in the payload and applies various matching techniques for hash tables and regular expressions. In encrypted traffic, DPI can be conducted by identifying plain text in the upper header of the payload or analyzing the hash values. Lin et al. [14] present string matching as a very useful technique in DPI that can be used for intrusion detection, malware scanning, and content filtering. Automaton-based, heuristic-based, and filtering-based methods have been used for string matching. Automaton-based algorithms find partially matched patterns in the text by changing its position, while heuristic-based algorithms skip certain characters to speed up the string search; they determine whether a suspicious match has occurred and move onto the next window position if has not. Filtering-based algorithms search the text for necessary pattern features and quickly exclude content that does not contain these features [14]. However, DPI analysis is not ideal for network security applications because of its vulnerability to algorithmic attacks. Moreover, because string-matching algorithms require significant computational power, high-performance hardware systems are required [11]. Another issue is that because the network administrator cannot control encrypted communications such as SSH and SSL/TLS, DPI is not useful for encrypted communications. For AET obfuscated with abnormal encryption, string-matching detection can be frustrated by concealing the payload [18].

3.3 Machine-learning-based analysis

DPI is useful for analyzing known network protocols, but it has limitations in analyzing encrypted traffic. When using ML, it is common to classify network communication based on supervised learning. Representative classification algorithms used for this purpose are the Naïve Bayesian, support vector machine (SVM), convolutional neural network (CNN), and decision tree algorithms.

The Naïve Bayesian algorithm is based on the probability for each feature and assumes independence between the features by default. It is widely used in a supervised learning environment and offers good performance even with little training data.

The SVM classification algorithm is commonly used for medium-sized datasets and conducts classification based on the decision hyperplane that provides the largest margin.

In addition, the CNN algorithm is a deep-learning algorithm that is often used to find characteristic patterns in images. This approach repeats the convolution layer and the pooling layer to extract and learn the characteristics of a given dataset. When repeating convolution and pooling, the number of hidden layers increases, and the process of analysis is not interpretable.

The decision tree algorithm is based on classification and regression trees. It produces a linearized tree, and the outcome of the leaf nodes depends on the decision rules [17]. The decision tree algorithms offer simplicity when interpreting models. It also considers the most important factors in a dataset first by placing them at the top of the tree [25]. The decision tree algorithm can effectively map nonlinear relationships. Tree-based algorithms lead to predictive models with high accuracy, stability, and ease of interpretation [7].

In Alshammari and Zincir-Heywood's work, decision tree, Naïve Bayesian, and SVM algorithms were used to classify SSH and non-SSH communications and their DR and FPR assessed. The highest DR and lowest FPR were produced in the predictive model using C4.5, a decision tree algorithm [3]. We thus employed a combination of the decision tree algorithm, which is widely used to classify network applications [3,4,6,8], and the Random Forest algorithm [5,20,26], which is employed to construct Decision Trees.

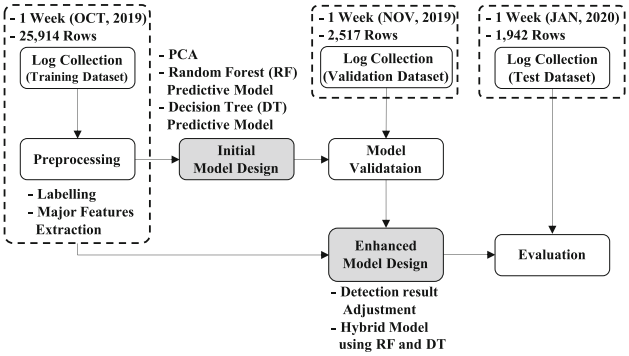


Fig. 2 SSH predictive model design process

Table 3 *cs_uri_port* and *cs_user_agent* based LABEL: Some application can be accurately classified by referring to the application information provided by the *cs_user_agent*

cs_uri_port	cs_user_ agent (Regular Exp.)	LABEL
22, 2022	–	SSH
–	^Mozilla/5.0*	Web
–	^Apache-HttpClient/4*	Web
–	^Dalvik/2.*	Mobile APP
–	^aws-sdk-dotnet-45/.*	Dev APP
–	^Zeplin/2.*	Dev APP

4 Methodology

4.1 Overview

In this section, we summarize the SSH predictive model used in the present study (Fig. 2). It consists of the following steps:

1. *Log Collection* HTTP CONNECT logs including SSH communications are collected from the web proxy.
2. *Preprocessing* Labels are added to the initial data sets used for supervised learning, such as decision tree and random forest. Besides, the data of features are converted into normalized numeric values.
3. *Initail Model Design* Major Features and PC1 (optimal principal component) are used for supervised learning using decision tree and random forest. In the PCA, features except for major features are simplified with keeping characteristics of the dataset to reduce the dimension of the dataset.
4. *Model Validation and Enhanced Model Design* The optimized predictive model is designed by analyzing the prediction result of the collected new dataset, unlike the training dataset. The DR and FPR of the last predictive model are evaluated using another dataset collected over different periods.

4.2 Web proxy log collection

We collected a dataset from the web proxy (Symantec ProxySG) installed on the Internet access barrier. The web proxy logs were extracted from the actual network environment of a company in Seoul, Korea (10,000+ employees), and the collection period was 1 week (October 18–24, 2019). A total of 13,187,275 HTTP CONNECT logs were collected. Communications without a server response and internal enterprise communications were excluded. The total number of rows of the logs were reduced to 25,914 except for TCP_ACCELERATED communications, which occur after the successful handoff of TCP_TUNNELED. For heuristic analysis, logs can be used without additional conversion. However, to apply the raw logs to the ML model, the information should be converted to numerical values. PCA can be applied to reduce log features during preprocessing, though it must be normalized in order to analyze the correlations.

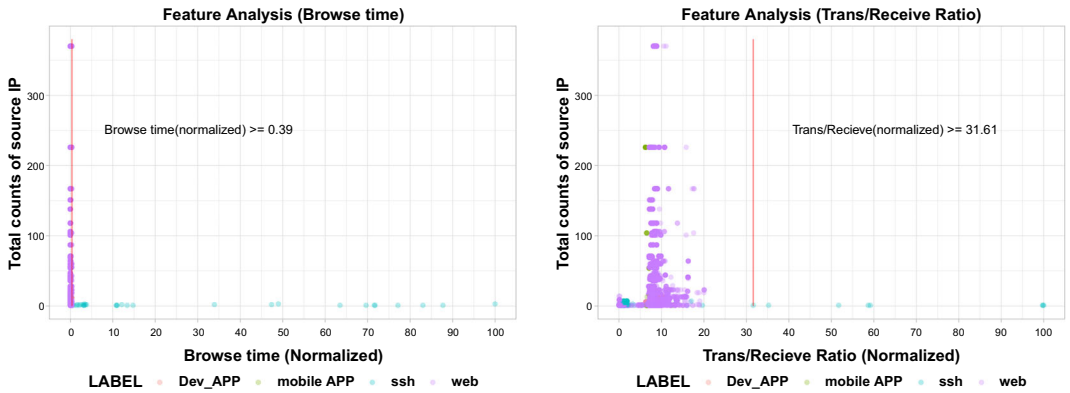
We set the labels for supervised learning by identifying the service attributes of web proxy logs. First, we classified SSH communications based on the destination port. TCP port 22, commonly used for SSH communications, and TCP port 2022, used for SSH communications within the environment from which the logs were collected, were labeled as SSH. Next, Web, Mobile APP, and Dev APP communications, which account for a high proportion of all communications, were classified based on their *cs_user_agent* information shown in Table 3. When considering that the detection model predicts unknown SSH traffic on the actual network, we selected logs that did not provide the web proxy's site category.

Table 4 summarizes the features of the dataset. *Original Dataset* represents the log content obtained directly from the web proxy, while *Preprocessed Dataset* is a feature used for the training of the prediction model. Part of the *Preprocessed Dataset* was not provided directly by the web proxy session logs but was derived from the analysis of cross-features and cross-rows. Features such as *log_count_connect_IP*, *log_count_total_connect*, and *log_avg_count_connect* were associated with the counted number of source IPs to the same destination server. The feature *log_transmit_speed_BPS* was obtained by dividing *cs_byte* by *browse_time*. *date_time*, *cs_host*, and *r_ip*, which are not numerical values, were transformed into binary numbers. The other features were provided as numerical values, but their variance was large. Therefore, we used log transform and normalization (0–100) on these features.

We analyzed and visualized the main attributes of SSH communications before constructing the model to detect SSH communications using ML algorithms. In particular, we determined whether SSH communications exhibited distinct patterns in terms of *browse_time* and *ratio_trans_receive*.

Table 4 Preprocessed dataset: The data converted from the DARPA99 dataset into the preprocessed dataset can be downloaded from "https://github.com/junimirang/SSH-Predictive-Model"

Original dataset	Preprocessed dataset	Value description
date_time	Business.time (1or0)	07:00–19:00 (1)
		20:00–06:00 (0)
browse_time	log_time_taken	
c_ip	–	
sc_byte	log_ratio_trans_receive	Value of cs_byte/sc_byte
cs_byte	log_cs_byte	
cs_host, r_ip	no_url (1 or 0)	if (cs_host=r_ip)
		Then no_url←1
		*If the communication has no url address, the value of no_url is '1'
cs_uri_port, cs_user_agent	LABEL	Web, SSH, Mobile_APP, Dev_APP
sc_category	–	Unknown destination ip has no category
cs_method	–	No info. appeared in the HTTP CONNECT
–	log_count_connect_IP	Number of c_ip connected to the same cs_host
–	log_count_total_connect	Number of connections to the same cs_host
–	log_avg_count_connect	Average number of connections per IP to the same cs_host
–	log_transmit_speed_BPS	Average transfer speed



(a) Browse time **(b)** Ratio of transferring and receiving traffic

Fig. 3 Major feature analysis

We also produced scatter plots using the preprocessed log values to identify features that could be used to distinguish between application types. We expected that *browse_time* and *ratio_trans_receive*, which are similar to the flow-based features used in Alshammari and Zincir-Heywood's work [4], would have a high correlation with the communication protocol. For the analysis of the web proxy logs, we used the ***R*** program.

Figure 3a, b shows that *browse time* and *trans/receive ratio*, respectively, provide a threshold for separating SSH from non-SSH traffic. In particular, as shown in these scatterplots, only SSH communications have a *browse time* over 618 s (normalized value: 0.39) and a *trans/receive ratio* (Transfer Byte/Receive Byte) over 22.78 (normalized value: 31.61). The *y*-axis represents the total number of source IPs connected to the same destination site with the same network protocol. This information can be used to predict whether the destination system is a public or private (or anomalous) site. If the number of source IPs is relatively small, we can predict that the destination system is not public.

4.3 Feature dimension compression using principal component analysis

Principal components analysis (PCA) is a widespread multivariate statistical technique, which is designed to extract meaningful information by simplifying a dataset while maintaining its characteristics and analyzing its structure. The primary goal of PCA is to find an axis that preserves the variance of the original dataset as much as possible and reduces

the dimensions by projecting the data onto that axis. In other words, when projecting the original dataset, we find the axis with the maximum variance. PCA has the eigenvector $\overrightarrow{e}$, which is the unit vector of n axes. We use PCA to visualize which features have the greatest impact, to improve the decision tree's performance from the collinearity between features, and to update of predictive modeling rapidly.

$$Var[X\overrightarrow{e}] = \frac{1}{m}\sum_{i=1}^{m}[X\overrightarrow{e} - E(X\overrightarrow{e})]^2 \quad (1)$$

$$= \overrightarrow{e}^T\left(\frac{X^TX}{m}\right)\overrightarrow{e}, \quad (C = \frac{X^TX}{m}) \quad (2)$$

$$= \overrightarrow{e}^T C\overrightarrow{e} \quad (3)$$

PCA seeks to find the eigenvector that maximizes $Var[X\overrightarrow{e}]$, a process that consists of the following steps:

1. Find the axis with the largest variance in the training dataset.
2. Find the second axis that is orthogonal to the first axis and has the largest variance.
3. Find the third axis orthogonal to the first and second axes and preserve the variance as much as possible.
4. Continue until the nth (the number of features) axis is found .

The unit vector defining the nth axis is the nth principal component (PC).

The eigenvalue associated with a component is equal to the sum of the squared factor scores for this component. The ratio of the squared factor score to the components is called *ctr* (contribution of the observation to the component) [23]. If *ctr* close to 1, the simplified dataset can maintain its characteristics as much as possible.

$$ctr = \frac{n\text{th eigenvalue}}{\sum \text{eigenvalue}} \quad (4)$$

In the preprocessed dataset, the dimensions of all log features except for *browse_time*, *ratio_trans_receive*, and *no_url*, were reduced by projecting six dimensions onto a single dimension using PCA. The eigenvalues and the eigenvectors were obtained using the "numpy" library from Python. When processing a total of 25,914 rows, 87.15% (*ctr* = 0.8715) of the data was maintained, and the dimensionality was reduced. Figure 4 presents the results of the PCA analysis, including the relationship between browse_time and *ratio_trans_receive*. SSH communications were concentrated from 0.026 to 1.30 for PC1, while web communications had a wider distribution (0.0–43.0). The PCA confirmed that

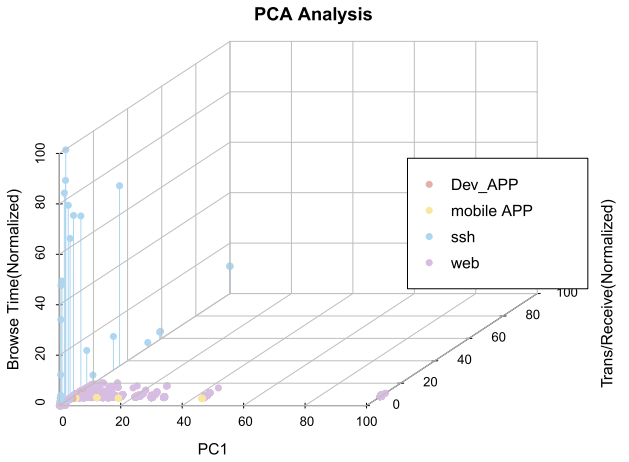


Fig. 4 Scatter plot for the major and principal components: The first PC with the highest ctr is also related to SSH classification

the SSH characteristics show the relation of the other features except for *browse_time* and *ratio_trans_receive*.

4.4 SSH detection model design

4.4.1 Decision tree and random forest algorithms

In this paper, we used the decision tree and random forest algorithms for the ML detection model. The decision tree algorithm generates a set of predictable rules by repeating the classification based on data features. Classification proceeds in the direction of increasing homogeneity and decreasing impurity or uncertainty as much as possible in each domain. Decision trees provide high prediction performance compared to their computational complexity. In Eqs. 5 and 6, R_i is the proportion of records belonging to area i. As the classification method, entropy and the Gini index are most commonly employed. In the proposed model, we use the Gini index for the decision tree.

$$\text{Entropy}(A) = \sum_{i=1}^{D} R_i\left(-\sum_{k=1}^{m} P_k \log_2 P_k\right) \quad (5)$$

$$Gini\_Index = \sum_{i=1}^{D} R_i\left(1 - \sum_{k=1}^{m} P_{ik}^2\right) \quad (6)$$

The random forest algorithm solves the problem of the overfitting of the decision tree. The random forest algorithm creates multiple decision trees and passes new data through each tree at the same time. The final result is selected using the average of the results. A sample dataset is obtained from the original dataset through bootstrap aggregating (bagging). When selecting the sample dataset from the original dataset, bagging allows the duplicate selection of data. In our study, the random forest algorithm consisted of 100 decision trees,

Table 5 Comparison of the test results with the training datasets: All detection algorithms using web proxy session log or network packet flow show similar detection rates in the training dataset

Model	Detection rate (Recall)	Detection algorithm	Dataset type
	0.99	Random forest	Web Proxy Session Log (employee 10,000 + company)
Proposed Model	0.99	Decision tree	
	0.97	Random forest	Network Packet Flow to Session Log Form (DARPA99)
	0.96	Decision tree	
Alshammari et al. [4]	0.99	*SBB-GP	Network Packet Flow (Dalhousie Univ.)
Vinayakumar et al. [21]	0.98–0.99	RNN	Network Packet Flow (MAWI/AMP/NIMS)
Lotfollahi et al. [15]	0.98	CNN	Network Packet Flow (ISCXVPN2016)

*SBB-GP, Symbiotic Bid-based Genetic Programming

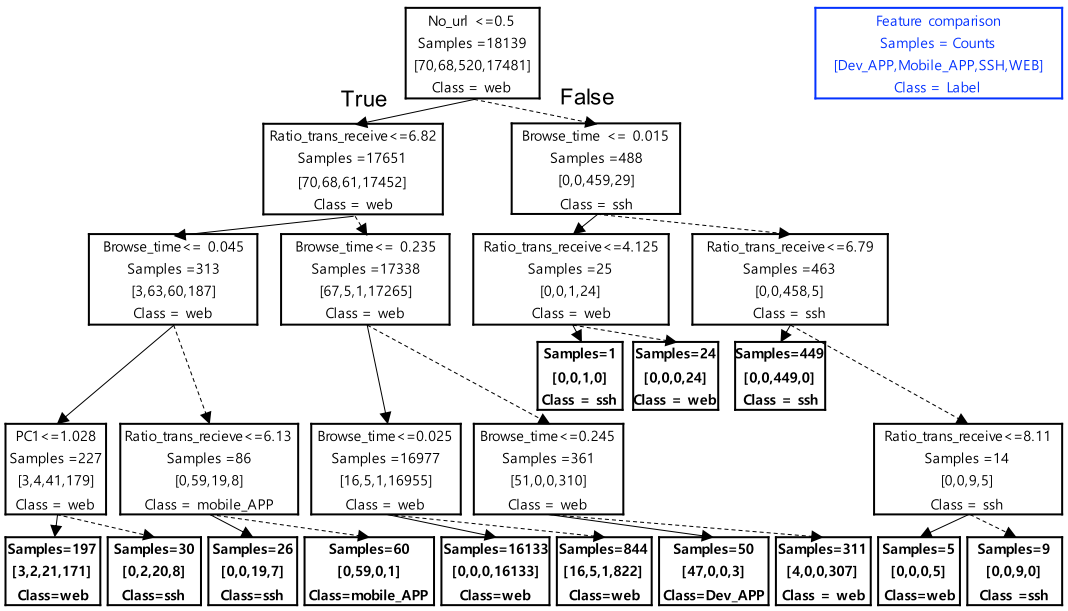


Fig. 5 Diagram of the decision tree predictive model (depth = 4)

and the depth was not limited until all leaves became pure or fell to less than 2.

4.4.2 SSH traffic prediction using machine learning

A predictive model using labeled sample data is required to confirm the communication properties of newly generated unclassified traffic. First, we predicted unknown communication properties using the random forest algorithm, which is an ensemble learning method. Features such as *browse_time*, *ratio_trans_receive*, *no_url*, and *PC* have been used in the decision-making process to classify communication attributes. We used the scikit-learn module of Python for the random forest algorithm and the decision tree algorithm. In the decision tree algorithm provided by scikit-learn, the optimized version of the CART algorithm is used.

For supervised learning, the 25,914 rows of training dataset were divided, at a 7:3 ratio, into an 18,139-row training samples and a 7,775-row test samples. After 100 training cycles, a predictive model using the random forest algorithm, which has 100 estimation counts, produced an average accuracy of 99.9% for the 7775-row test samples. This level of accuracy remained the same for 1000 estimates.

Our prediction model used web proxy session logs, which are smaller in size than network packet flows. In Table 5, the proposed model produced a similar performance to prediction models that employ network packet flows. In general, the predictive model using the random forest algorithm generates a high DR. However, the random forest algorithm is not transparent, making it difficult to determine which features affected the decision. We thus tested the predictive model using the decision tree algorithm to determine the important features in this classification.

The decision tree diagram in Fig. 5 was employed to identify key features used to predict communication properties. This decision tree model had a maximum depth of 4 and

had a 99.52% detection accuracy when using the same training dataset as the random forest algorithm. Figure 5 shows that *no_url*, *ratio_trans_receive*, and *browse_time* were the important features when classifying communication properties.

We prepared a validation dataset that was not used in training the model to verify its predictions. We collected logs for a week (November 18–25, 2019) from the web proxy in the same network section as before. The accuracy of the predictive model was measured using 2517 rows from the web proxy, which did not categorize the internet sites and which had a label. The 2517 new data used for prediction consisted of 1161 web communications and 1356 SSH communications. Figure 6 presents the results of evaluating the new dataset based on 100 predictive cycles of the model. The predictive model using the random forest algorithm averaged 60.44% for accuracy, while that using the decision tree algorithm averaged 45.26%. In the test using the validation dataset, the DR was significantly lower than the detection accuracy of the training dataset. Because only a few communication sessions with the same SSH server were detected as SSH communications, the DR for each row was low.

The predictive model should consider the prediction results of other rows to increase the DR of SSH communications. Unlike HTTP/HTTPS servers, an SSH server provides one service on the same destination IP and port. Unless the TCP port assigned to the SSH service is changed, the communications are forwarded to the same IP and port. Thus, if a certain row in multiple logs that is forwarded to the same destination IP and port is an SSH communication, we can also conclude that the other communications are SSH communications.

As seen in the third row of Table 6, communications forwarded to "115.68.14.237:2022" had different prediction counts. However, because the three rows were predicted to be SSH communications, we can change the other rows to SSH communications under the premise that the prediction of SSH communications is stronger than that of other applications.

4.4.3 Detection adjustment considering SSH properties

In the 2517 rows from the web proxy, we identified 155 servers based on the destination IP and port. By repeating the generation of the predictive model, communications from 155 servers were predicted. As a result, the average DR of the random forest predictive model was 57.98% and that of the decision tree predictive model was 60.55%. Figure 6 shows the DR and FPR of two predictive models for SSH communications. The decision tree predictive model has a relatively high DR for SSH communications. Table 7 presents the detection results for the highest SSH detection accuracy from the decision tree predictive model.

When predicting a service based on the destination IP and port, the decision tree predictive model sometimes predicts SSH servers that cannot be detected by the random forest prediction model and sometimes has a higher DR for SSH communications than the random forest predictive model. However, in the decision tree predictive model (Fig. 6), the FPR was relatively high, and the standard deviation was large. Thus, when using the decision tree model for the prediction of SSH communications, the FPR should be considered first.

4.4.4 Design of a hybrid predictive model

To maintain the high DR of the decision tree predictive model and to decrease the FPR, we designed a hybrid predictive model that combines the random forest and the decision tree predictive models. When we repeated the predictive model test, we confirmed that the root node of the decision tree model is an important factor in decreasing the FPR. We designed the hybrid predictive model to minimize the FPR for SSH communications detected in the decision tree using the root node (*no_url*). Algorithm 1 is pseudo-code showing the process of combining the two predictive models.

Algorithm 1: Hybrid predictive model

```
if D_RandomForest = SSH then
  | D_Hybrid ← SSH
else
  | if D_DecisionTree = SSH and no_url = 1 then
  |   | D_Hybrid ← SSH
  | else
  |   | D_Hybrid ← D_RandomForest
  | end
end
```

In Fig. 6, we employed a new dataset to process the logic designed to improve the DR and FPR of the hybrid predictive model. We examined the predictive models for cases with the highest DR of the 100 repeated estimations of the prediction model. In the seventh case in Fig. 6, the decision tree model detected 19 more SSH communications than did the random forest model. In the hybrid predictive model, Algorithm 1 is applied to the detection results for the random forest model, and 14 SSH communications were detected while minimizing the FPR (Table 8).

5 Evaluation

We collected test dataset from January 6–12, 2020, to evaluate the hybrid predictive model. A total of 151,060 rows of HTTP CONNECT session logs were collected under the same conditions as for the datasets used to construct the predictive models. Of these rows, 1942 classified with a label

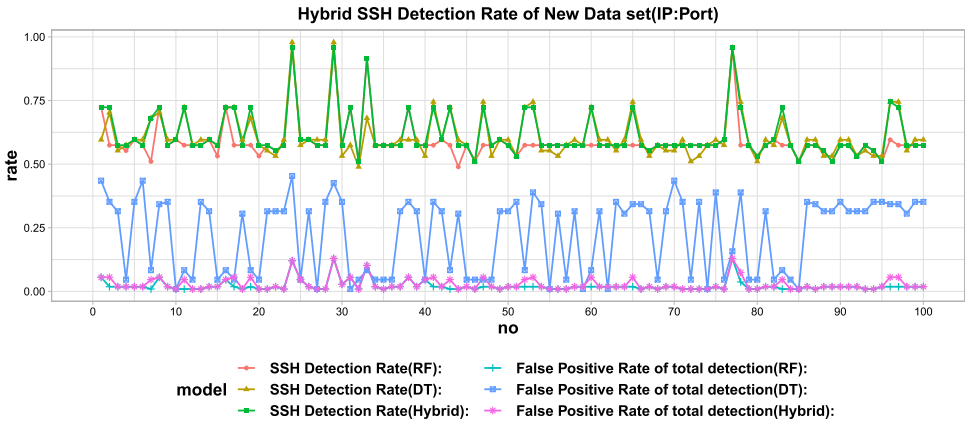


Fig. 6 DR and FPR of the predictive model for SSH communications

Table 6 Predictions considering the destination IP and port

Destination.ip	cs.uri.port	Row count of dt_ssh	Row count of dt_web	Final Prediction
112.106.161.127	443	–	338	Web
112.106.189.163	447	2	–	SSH
115.68.14.237	2022	3	279	SSH

Table 7 Predictions using the random forest and decision tree algorithms (best row detection accuracy case)

Application classification	ML prediction using decision tree				
	Decision tree		Random forest		Total sum
	SSH	Web	SSH	Web	
SSH	46 (TP)	1 (FN)	45 (TP)	2 (FN)	47
Web	49 (FP)	59 (TN)	13 (FP)	95 (TN)	108
Total sum	95	60	58	97	155

were prepared. Table 9 presents the results of the hybrid model, with a recall of 70.52% and an FPR of 5.18% for 23 SSH servers corresponding to 1153 rows. The evaluation results were better than those of the individual models that made up the hybrid model.

We further evaluated the hybrid model with the DARPA 99 data set used by Alshammari et al. However, because the DARPA 99 dataset is in tcpdump format, which collects network packets, it was impossible to use it directly in our proposed model. Before adopting the DARPA 99 dataset, preprocessing was conducted to transform the dataset into a similar format to the web proxy session logs. We converted the dataset by extracting the metadata, including the TCP/IP header information from tcpdump, and merging this information into a TCP session [13].

In the DARPA 99 dataset, we used the network packet flows of the fourth week as the training dataset and the data of the first and third weeks as the test dataset. In the training dataset, only packets corresponding to TCP were extracted from the metadata of 6,461,795 packets. The training dataset of 177,218 sessions, while the test dataset consisted of 463,908 sessions generated from 14,604,905 packets. However, because the DARPA 99 dataset was collected in 1999, when SSH traffic volumes were low, we required additional SSH data. According to Alshammari's study, if the percentage of SSH traffic across the entire dataset is low, Telnet, which has properties similar to SSH, can be used to predict SSH communications [2]. Therefore, we changed the labels for Telnet to SSH. As a result of measuring the performance using the DARPA 99 dataset, the recall of test dataset was 100%, the precision was 19.7%, and the FPR was 1.1% (Table 10). As a result, our proposed model outperformed Alshammari et al.'s model, which employed a Gaussian process [3].

When comparing the dataset of the ICT company network used in the modeling and the DARPA 99 dataset in Table 9 and 10, the recall of DARPA 99 increases and the precision of DARPA 99 decreases. We found that this difference is due to the ratio of the label in the test dataset (SSH's ratio of real network: 59.4%, SSH's ratio of DARPA 99: 0.34%). As the ratio of SSH (positive) in the evaluation data set decreases, false negative decreases and false positive increases. In this case, according to Eqs. 7 and 8, the recall for SSH detection increases and the precision decreases.

$$\text{Recall} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalseNegative}} \tag{7}$$

$$\text{Precision} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalsePositive}} \tag{8}$$

Table 8 The number of SSH servers only counted in D_{DT} (for the largest detection gap between the random forest and decision tree models)

Label	① $(D_{DT} = SSH\ AND\ D_{RF} \neq SSH)$		Total Sum
	no_url=0	② no_url=1	
SSH	1	11	12
Web	3	3	6
Mobile APP	1	0	1
Total Sum	5	③ 14	19

Table 9 Prediction accuracy using a test dataset (average of 100 runs): The hybrid predictive model improves the detection rate by using the prediction results from the decision tree algorithm and adjusting the FPR by referring to the presence of a URL

Predictive model	Detection rate (Recall) (%)	Precision (%)	FPR (%)	Dataset type
Random forest	62.8	87.5	3.8	Web proxy session log
Decision Tree	72.0	82.5	6.4	
Hybrid	**70.5**	**85.4**	**5.2**	

Bold expressions are used to emphasize performance comparison with competitors

Table 10 Evaluation using the first and third week traffic of DARPA 99 dataset as a test dataset (average of 100 runs): The hybrid predictive model had a recall of 100%, and an FPR of 1.1%, which were more favorable than Alshammari et al.'s model (*SBB-GP: Symbiotic Bid-based Genetic Programming "https://web.cs.dal.ca/~mheywood/Code/SBB/")

Predictive model	Detection rate (Recall) (%)	Precision (%)	FPR (%)	Dataset type
Random forest	88.1	36.3	0.4	Network packet flow (DARPA99)
Decision tree	98.9	20.3	1.0	
Hybrid	**100**	**19.7**	**1.1**	
Alshammari et al. (*SBB-GP)	**98.3**	**–**	**1.7**	Network packet flow (DARPA99)

Bold expressions are used to emphasize performance comparison with competitors

6 Conclusion and future work

This paper uses the application session logs of a web proxy to detect SSH communications, which differs from previous research [3,4] based on network flow datasets. The use of web proxy's session logs can save time and computational resources in the gathering and processing of packet flow data for network flows, while the additional features (e.g., cs_host, cs_user_agent) provided by a web proxy can be used for ML analysis. This paper also identified the features of web proxy logs to identify SSH communications and explained the process for using these features in ML. We proposed preprocessing that vectorized the log features and reduced the feature dimensions for practical ML analysis. By presenting this analysis methodology based on ML, it is expected that we can use web proxy logs for various purposes in the future.

The datasets used in the design and verification of the predictive models in this paper consisted of logs collected from the network environment of an actual company. Though the characteristics of the collected dataset may differ according to the characteristics of this company, the proposed methodology is useful for companies using a web proxy in general. The detection performance can be improved by collecting logs within the actual network environment and updating the predictive model. In future research, we will propose a model that classifies SSH traffic details (command script transfer, file upload, file download, reverse connection) to minimize false-positive detection and detect anomalous traffic more accurately. We will also continue to perform the classification and labeling of communications to acquire the various applications required for supervised learning.

Acknowledgements This work is supported in part by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2019-0-01343 Regional strategic industry convergence security core talent training business, No. 2019-0-01697 Development of Automated Vulnerability Discovery Technologies for Blockchain Platform Security, and No. 2020-0-01819 ICT Creative Consilience program).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Informed consent was obtained from all individual participants included in the study.

References

1. Art. 25 GDPR—Data protection by design and by default. https://gdpr-info.eu/art-25-gdpr/
2. Alshammari, R., Zincir-Heywood, A.N.: A flow based approach for SSH traffic detection. In: 2007 IEEE International Conference on Systems, Man and Cybernetics, IEEE, pp. 296–301 (2007)
3. Alshammari, R., Zincir-Heywood, A.N.: Machine learning based encrypted traffic classification: Identifying SSH and skype. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, IEEE, pp. 1–8 (2009)
4. Alshammari, R., Zincir-Heywood, A.N.: Can encrypted traffic be identified without port numbers, ip addresses and payload inspection? Comput. Netw. **55**(6), 1326–1350 (2011)
5. Bagui, S., Fang, X., Kalaimannan, E., Bagui, S.C., Sheehan, J.: Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features. J. Cyber Secur. Technol. **1**(2), 108–126 (2017)
6. Boutaba, R., Salahuddin, M.A., Limam, N., Ayoubi, S., Shahriar, N., Estrada-Solano, F., Caicedo, O.M.: A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. J. Internet Serv. Appl. **9**(1), 16 (2018)
7. Brid, R.S.: Decision trees—a simple way to visualize a decision (2018). https://medium.com/greyatom/decision-trees-a-simple-way-to-visualize-a-decision-dc506a403aeb
8. Bujlow, T., Riaz, T., Pedersen, J.M.: A method for classification of network traffic based on c5. 0 machine learning algorithm. In: 2012 International Conference on Computing, Networking and Communications (ICNC), IEEE, pp. 237–241 (2012)
9. Cai, T., Zou, F.: Detecting http botnet with clustering network traffic. In: 2012 8th International Conference on Wireless Communications, Networking and Mobile Computing, IEEE, pp. 1–7 (2012)
10. Chammem, M., Hamdi, M., Kim, T.H.: Extending advanced evasion techniques using combinatorial search. In: 2014 7th International Conference on Security Technology, IEEE, pp. 41–46 (2014)
11. Dharmapurikar, S., Krishnamurthy, P., Sproull, T., Lockwood, J.: Deep packet inspection using parallel bloom filters. In: Proceedings of the 11th Symposium on High Performance Interconnects, 2003, IEEE, pp. 44–51 (2003)
12. Ferrara, P., Spoto, F.: Static analysis for GDPR compliance. In: ITASEC (2018)
13. Flow2session. https://github.com/junimirang/Flow2Session
14. Lin, P.C., Lin, Y.D., Lai, Y.C., Lee, T.H.: Using string matching for deep packet inspection. Computer **41**(4), 23–28 (2008)
15. Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R., Mohammdsadegh, S.: Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft. Comput. **24**, 1999–2012 (2020)
16. Marty, R.: Applied Security Visualization. Addison-Wesley, Upper Saddle River (2009)
17. Mighan, S.N., Kahani, M.: A novel scalable intrusion detection system based on deep learning. Int. J. Inf. Secur. (2020). https://doi.org/10.1007/s10207-020-00508-5
18. Neupane, K., Haddad, R., Chen, L.: Next generation firewall for network security: a survey. In: SoutheastCon 2018, IEEE, pp. 1–6 (2018)
19. Shah, A., Banakar, V., Shastri, S., Wasserman, M., Chidambaram, V.: Analyzing the impact of {GDPR} on storage systems. In: 11th {USENIX} Workshop on Hot Topics in Storage and File Systems (HotStorage 19) (2019)
20. Shen, M., Zhang, J., Chen, S., Liu, Y., Zhu, L.: Machine learning classification on traffic of secondary encryption. In: 2019 IEEE Global Communications Conference (GLOBECOM), IEEE, pp. 1–6 (2019)
21. Vinayakumar, R., Soman, KP., Poornachandran, Prabaharan.: Secure shell (ssh) traffic analysis with flow based features using shallow and deep networks. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, pp. 2026–2032 (2017)
22. Wagener, G., Dulaunoy, A., Engel, T.: Towards an estimation of the accuracy of TCP reassembly in network forensics. In: 2008 Second International Conference on Future Generation Communication and Networking, IEEE, vol. 2, pp. 273–278 (2008)
23. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemom. Intell. Lab. Syst. **2**(1–3), 37–52 (1987)
24. Wullink, M., Moura, G.C., Müller, M., Hesselman, C.: Entrada: a high-performance network traffic data streaming warehouse. In: NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium, IEEE, pp. 913–918 (2016)
25. Xhemali, D., Hinde, J.C., Stone, G.R.: Naïve bayes vs. decision trees vs. neural networks in the classification of training web pages. Int. J. Comput. Sci. Issues **4**(1), 16–23 (2009)
26. Yamansavascilar, B., Guvensan, M.A., Yavuz, A.G., Karsligil, M.E.: Application identification via network traffic classification. In: 2017 International Conference on Computing, Networking and Communications (ICNC), IEEE, pp. 843–848 (2017)
27. Yang, W., Cheng, Z., Cui, B.: Recombining TCP sessions based on finite state machine to detect cyber attackers. In: Proceedings of the 3rd International Conference on Cryptography, Security and Privacy, pp. 138–142 (2019)
28. Yoon, S.H., Park, J.W., Park, J.S., Oh, Y.S., Kim, M.S.: Internet application traffic classification using fixed ip-port. In: Asia-Pacific Network Operations and Management Symposium, Springer, pp. 21–30 (2009)