Classification of Freenet Traffic Flow Based on Machine Learning

Seungwoon Lee[1], Seung-hun Shin[2], and Byeong-hee Roh[1]
[1] Dept. of Computer Engineering, Ajou University, Suwon 16499, Korea
[2] Dasan University College, Ajou University, Suwon 16499, Korea
Email: {swleeyg, sihnsh, bhroh}@ajou.ac.kr

Abstract —An anonymous overlay network is a virtual and logical network which can assure privacy but is often misused as a crime. Therefore, it is necessary to find users operating the abnormal overlay network in managed network for network administrator. However, there is a lack of research on host detection using Freenet which is one of the popular anonymous overlay networks and all of previous methods require that at least one Freenet node be inserted into the network. In this paper, we propose classification of Freenet traffic flow based on machine learning. Through this, it is possible to identify the host operating the Freenet inside the network without joining Freenet. We also evaluate the performance of classification algorithms. Among them, Decision Tree is most effective with 94% of precision and 0.0029 sec of time spent.

Index Terms—Freenet, anonymous overlay network, traffic classification, network security

I. Introduction

An anonymous overlay network is a virtual and logical network which cannot be accessed via typical internet browsers. Commonly used anonymous overlay network services include Freenet [1], Tor [2], and I2P [3] were designed to be difficult to track with using non-standard protocols, ensuring anonymity. Because those networks provide strong anonymity, the number of users is gradually increasing and several software companies such as Facebook and Firefox have begun to support them [4], [5].

A problem with an anonymous overlay network is that there are many cases to abuse of anonymity and especially, there are black markets in which malicious software, illegal contents, drugs, guns are traded. Because anonymous overlay networks are not accessible through regular search engines like Google, it is easy to avoid tracking from the government. For this reason, anonymous overlay networks are also known as Darknet. For example, the most threatening Ransomware in recent years, WannaCry includes a module to install Tor for trading Bitcoin [6].

From a network administrator's point of view, it is necessary to find and respond to users who operate the abnormal overlay network in the network under management. This is different from individuals who focus on privacy. Especially for organization, misuse of an anonymous overlay network can lead to security problems and also cause traffic loads. In this case, the use of an anonymous overlay network should be detected.

Unlike Tor, there is a lack of research on Freenet that we are focusing in this paper. Baumeister *et al*. [7] and Tian *et al*. [8]-[10] proposed an information tracking attack using the vulnerability of the Freenet protocol. Roos *et al*. [11] proposed statistical method to measure the scale of Freenet, and Levin *et al*. [12] passive technique for detecting Freenet downloaders. However, all of these methods require that at least one Freenet node be inserted into the network. When the device becomes a Freenet node, its storage is used for file sharing. The storage stores encrypted files that may be used to commit a crime. It is needed to discover Freenet nodes without joining. These methods also target the node already in Freenet. This is unnecessary for the network administrator who want to find the Freenet user in managed network.

In this paper, we propose Freenet traffic flow classification based on machine learning. The network administrator monitors whole flows on the border router. These flows are filtered according to the nature of Freenet. The filtered flows are finally judged whether it is Freenet traffic flow or not by the classification algorithm. Using this method, only monitoring traffic dump can detect that the host is running the Freenet without joining the Freenet network.

The contributions of this paper include the following:

- Freenet node detection without joining Freenet
- Tailoring method for Classification of Freenet traffic flow from a set of captured traffic
- Optimized machine learning algorithm for real-time Freenet detection

The remainder of this paper is organized as follows. In Section II, we introduce Freenet briefly and review the related works. In Section III, we analyze the characteristics of Freenet traffic flow and propose the Classification of Freenet traffic flow based on Machine Learning. We evaluate the performance of classification method in Section IV, and conclude this paper and discuss our future work in Section V.

Manuscript received March 20, 2018; revised October 22, 2018.
Corresponding author email: bhroh@ajou.ac.kr
doi:10.12720/jcm.13.11.654-660

II. Background

A. Freenet

Freenet, developed by Ian Clarke, is one of the representative services of anonymous overlay networks. It can be regarded as a decentralized data repository for sharing information against censorship. Basically, Freenet node can publish and subscribe anonymously with Key based Peer-to-Peer (P2P) method. Without key, it is unable to obtain the target data. Here are two routing features why it is difficult to detect Freenet traffic flows.

Location Value Routing in Freenet uses their own feature called location value rather than a shortest path algorithm used in a typical network. Location value is specified by hashing the XOR operation value with various values such as the position value and seed value of the neighbors. Freenet nodes act as logical routers as well as senders and receivers. When a node receives a file request from a neighbor node, it relays the request message to the neighbors while hiding the request information like the onion routing (Tor). At this time, the route to be forwarded is determined by the location value not the IP address of the destination. The node also periodically changes this value to avoid tracking.

Key-Based Routing Another routing feature of Freenet is key-based data transmission. In Freenet, all content can be represented by a hashed key value. The hash value of the content is also converted to a location value and stored in the node with the closest location value. Because of the hash function for which the result is completely different even if the data are slightly different, the key value is completely different even for similar files. Due to this, when similar files are stored on a Freenet, it can automatically prevent data from being stored in the same or adjacent nodes. Nodes use keys when requesting data. The request node asks its neighbor who has the closest location value to the location value of the key to see if it has the data. The data is transmitted if there is the data in it. In the contrast, neighbor asks its neighbor whether it has file if there is no data. As this process is repeated, the request message is delivered to the node that has the data, and the file is delivered to the requesting node along the path the message was sent.

The characteristics at the packet level are as follows.

UDP Freenet nodes send packets with UDP (User Datagram Protocol) as transport layer protocol. Freenet packet includes an IPv4 header of 20 bytes and a UDP header of 8 bytes. (IPv6 is also available but not considered in this paper). For this reason, the applications such as UDP, VoIP and BitTorrent are compared with Freenet.

Packet Length There is a padding value of 0-127 Bytes at the end of the packet to apply as the MTU (Maximum Transmission Unit) value. MTU is the size of data that can be transmitted (Byte). Maximum packet size can be limit the by setting the MTU value in Freenet Configuration and the default MTU is 1280, the minimum is 576. Even it is configurable, the user does not change the default value in general.

Port Number Randomization Freenet has been assigned the port randomly during installation and supports manual change of port number. This prevent Freenet node from tracking and detection using the port number. That is, the port number cannot be used as a function of classification.

B. Related Work

Several studies have been conducted to collect information on Freenet. Baumeister *et al*. [7] proposed a Routing Table Insertion attack. Through this attack, the neighbor of the target node can be configured as the attacker's node. If all neighbor nodes of the target are attackers, the anonymity of the node is not guaranteed.

Tian *et al*. [8]-[10] proposed a traceable vulnerability and a countermeasure against UID. UIDs uniquely assigned to each message are stored for each node so that no loop is generated. However, in the earlier version of Freenet, the UID changes in the node where the data exists in the node transmitting the data back. Therefore, an attacker can assume that the node whose UID changes is the node holding the data. This is a factor that greatly hinders the anonymity provided by Freenet. The authors proposed a dynID (Dynamic ID) scheme to change the UID for each hop, and applied to the official update

Roos *et al*. [11] wanted to measure the scale of Freenet by inserting 80 nodes into the Freenet in 2014. These monitoring nodes logged all messages sent, received or relayed. They also measured the session length with the neighbors and the inter-session length. According to this experiment, the average number of online nodes connected to Freenet can be estimated as 2,000-3,000.

Levine *et al*. [12] proposed a passive technique for detecting Freenet downloaders. They derived a Bayesian framework for testing whether a peer may be downloading a document, based on counting the requests observed. This can determine if the node is the Original Requester or the relay node that requested the data. It requires only single node, but must own the manifests to monitor in advance.

In the studies above-mentioned, the detectors must join in the network as a Freenet node. When the detector joins Freenet, encrypted files that cannot be opened are stored in the detector's node and other nodes share the file via the detector. This may result in wasted resources at the detector's node, as well as engaging in illegal activities.

III. Classification of Freenet Traffic Flow Based on Machine Learning

A. System Overview

Fig. 1 shows the system overview of the proposed method. Packets from hosts on the internal network pass this border router in order to connect to the Internet and the classification system is able to monitor them on the border router. The host running Freenet exchange packets

with the external neighbor node. While the border router passes flow information to the connected classification system, the classification system can determine whether the flow is Freenet or not.

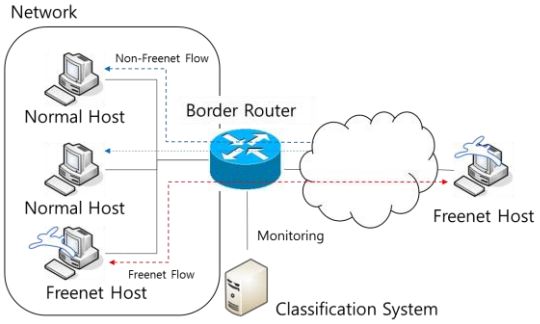


Fig. 1. System overview

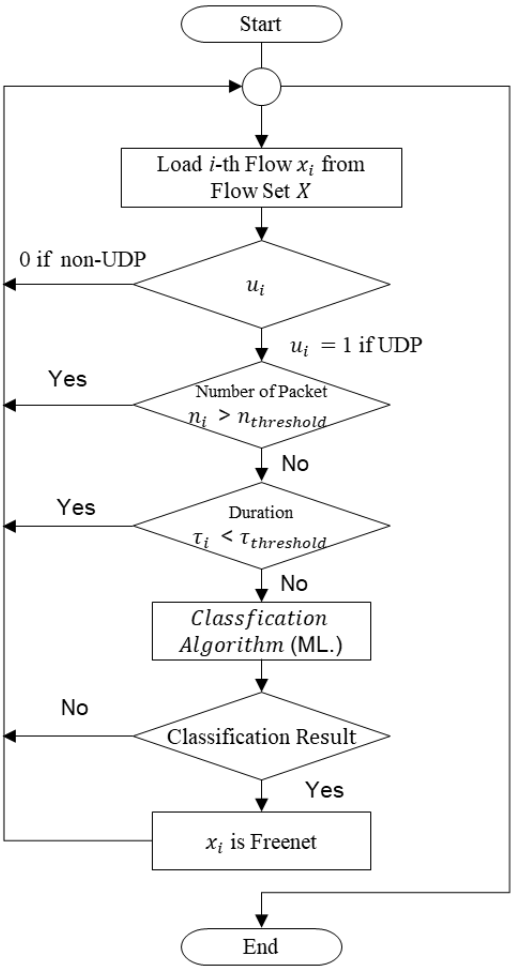


Fig. 2. Flow chart of classification procedure.

B. Classification Procedure

The proposed method assumes that the flow is collected by a border router. For ease of explanation, the symbols are defined as follows. X is a set of collected flows from the border router and the elements of X are represented by x_1, x_2, x_3,…, x_i. u_i is a Boolean value that is one when the transport layer protocol of x_i is UDP or zero when it is non-UDP. n_i is the number of packets in x_i and τ_i is the duration that is the difference between the arrival time of the last packet and the arrival time of the first packet in x_i. $n_{threshold}$ and $\tau_{threshold}$ are threshold value can be dynamically determined by user.

Follow the procedure below as shown in the flow chart in Fig. 2. First phase is that loading x_i from X. When it is done, Filtering phases are in progress step by step. x_i is filtered if u_i is zero (Non-UDP). If $n_i < n_{threshold}$ and if $\tau_i < \tau_{threshold}$, x_i is also filtered. Lastly, x_i is classified with Machine Learning Algorithm. When all phases are done, we can determine that flow is Freenet traffic flow.

C. Classification Algorithms

The machine learning algorithm to be used in the classification step can be dynamically varied according to the user's preference. However, selecting an algorithm that performs better classification can be more effective. The performance of each algorithm is evaluated in the next chapter. A description of a typical machine learning algorithm is as follows;

Logistic regression [13] is a probability model for regression analysis of a nonlinear model that is difficult to classify, and is mainly used when the dependent variable is a categorical variable. It is easy to implement and interpret, but there is a high probability that underfitting will work.

Support Vector Machine [14] finds hyperplanes that can classify multidimensional values and classify them based on them. Category, numerical prediction problem, etc., and is not greatly influenced by noise and excessive sum does not occur frequently. However, many parameter tests are needed to find the optimal model and training can be slow depending on the data set.

Decision Tree [15] is an analysis technique that classifies decision rules for items into trees. The results of the classification through the Decision Tree are expressed in a tree structure, which is advantageous for the analyst to easily understand and explain the results. Therefore, it can be used directly for decision making. However, because it is classified as a single variable, when the characteristics of data cannot be classified into specific variables, the classification rate is lowered and the tree becomes complicated.

Random Forest [16] is an algorithm to improve the disadvantages of Decision Tree. It creates one model by combining several decision trees which are slightly different by randomness. Because it guarantees diversity, it is superior in performance, but it cannot have intuition which is advantage of Decision Tree in analysis because it is decision method which is a combination of multiple trees.

K-Nearest Neighbors [17] is an algorithm to classify by referring to labels of k different data that are close to the data and can be implemented simply. In the numerical data classification task, although the performance is high, it is disadvantageous that the classification speed is

slowed down as the amount of training data is increased because the pre-calculation is not performed.

Multi-Layer Perceptron [18] is a kind of artificial neural network, which is an iterative optimization model that reduces the error rate by performing Feed-Forward on Regression several times. When the input data is forwarded to each unit of the input layer, it is converted in each unit, transferred to the hidden layer, finally outputted to the output layer, and the output value is compared with the expected value. The connection strength is adjusted again in the direction of reducing the difference (Feed). MLP is known as the most effective method for complex input, but it is highly volatile.

D. Classification Feature

Table I provides a list of the 7 features that we use to classify. Host information such as IP Addresses is not considered in classification but is only used for identity to prevent replication.

TABLE I: CLASSIFICATION FEATURES

Identity
A (SrcIP:Port)<->B (DstIP:Port)
Features
Total packets
Total Packet Size
Total Packets transmitted (A->B)
Total Packet Size transmitted (A->B)
Total Packets Received (B->A)
Total Packet Size Received (B->A)
Duration (Sec)

IV. EXPERIMENT

In this chapter, we experiment and evaluate how well the classification step of the proposed method will determine how effectively Freenet is categorized. We also measure the learning time and test time to assess the availability of real-time detection. The packet capture module was implemented in c ++, and the classification module was implemented in Python 3.5 using the Scikit-learn library and Pandas. Parameter $n_{threshold}$ and $\tau_{threshold}$ are set to 100.

TABLE II: COLLECTED APPLICATION IN FLOW DATASET

Classification		Application
Non-Freenet	VoIP	Skype, KakaoTalk VoiceChat, Discord
	Game	FIFA Online 3, League of The Legends, Grand Theft Auto 5, Tom Clancy's The Division, Dungeon and Fighters, Diablo 3, Overwatch, Insuergency, Starcrafts, PlayerUnknown's Battlegrounds
	P2P	Bit Torrent
	Others	DNS, ICMP, GVSP, RTCP ...
Freenet		Freenet

A. Dataset

We collected the experiment traces from single hosts. Table II lists the types of traffic dataset collected for validation. Typical applications that use UDP are classified as VoIP, game, P2P, etc. Since the proposed method is binary classification, those sets are integrated into the non-Freenet traffic flow set. The total number of flow collected is 2524 by excluding the duplicated items, and there are 398 Freenet traffic flows which are one-fifth of the total data. Classification categories and collection techniques are based on [19] which is previous researches related to UDP Traffic classification.

B. Classification Metrics

TABLE III: CONFUSION MATRIX

	True	**False**
Predict Positive	TP (True Positive)	FP (False Positive)
Predict Negative	TN (True Negative)	FN (False Negative)

In order to evaluate the classification algorithms, we used confusion Matrix, Precision, Recall, and Accuracy commonly used for evaluating existing machine learning algorithms. In Table III, True means that the entity and the predicted value match, and False means the case where the actual and predicted values do not match. Predict Positive and Predict Negative are predicted values. In this experiment, True Positive means the result of predicting Freenet as Freenet and True Negative means the result of predicting non-Freenet as non-Freenet. False Positive means the result of predicting non-Freenet as Freenet and False Negative means the result of predicting Freenet as non-Freenet.

Precision refers to the ratio of flows that are correct predicted among the flows predicted to Freenet traffic flow and Recall refers to the ratio of flows that predicted to Freenet traffic flow among the Freenet traffic flows. Accuracy is the ratio of accurately predicted results in total prediction results. They are defined as:

$$\text{Precision} = TP/(TP+FP) \quad (1)$$

$$\text{Recall} = TP/(TP+FN) \quad (2)$$

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN) \quad (3)$$

C. Validation

Monte-Carlo simulations were performed to validate the algorithm. The Monte-Carlo simulation test is a simulation method that randomly divides whole data into training set and test set and then repeats the test from hundred to thousand times. In the experiment, training set were randomly divided, and the average of the results was calculated after 1,000 repetitions. The ratio of training set to test set is 8:2.

D. Performance Evaluation

Figure 3 and Table IV show the average of the Precision, Recall, and Accuracy for the classification algorithms. Overall, the accuracy values are high,

because the percentage of non-Freenet traffic flows in the entire flow set is 80%, and even if none of Freenet traffic flow are detected, the accuracy result is more than 80%. Therefore, it is appropriate to focus on Precision.

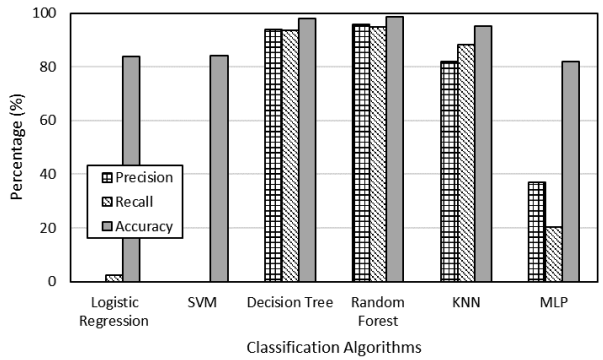


Fig. 3. Classification result of precision, recall, and accuracy.

TABLE IV: PRECISION, RECALL AND ACCURACY OF EACH ALGORITHM

(%)	Logistic Regression	SVM	Decision Tree	Random Forest	KNN	MLP
Precision	0	0	93.950	95.656	82.097	36.834
Recall	2.235	0	93.497	94.788	88.164	20.264
Accuracy	83.981	84.255	98.020	98.497	95.096	81.972

TABLE V: TRAINING TIME AND TESTING TIME OF EACH ALGORITHM

(sec)	Logistic Regression	SVM	Decision Tree	Random Forest	KNN	MLP
t_{Training}	0.006859	0.150273	0.002926	0.015139	0.001324	0.01935
t_{Testing}	0.000207	0.079762	**0.000228**	0.001516	0.003739	0.000443

Logistic Regression and Support Vector Machine work the worst with 0% of precision. They were unable to classify any Freenet traffic flows. Precision of Decision Tree and Random Forest were 93.95% and 95.66%, which were higher than 90% and Random Forest has a precision that is 1.5% higher on average than Decision Tree. They have recall and accuracy values above 90%. K-Nearest Neighbors has 82.1% of precision and Multi-Layer Perceptron has 36.83% of precision.

We also measure the time consumed to training and testing. Fig. 4 and Table V show the average of the training and testing time for the classification algorithms. The fastest algorithm for training was K-Nearest Neighbors by 0.001 sec and Decision Tree was followed by 0.0029 sec. Testing time is more important than training time because training phase is not performed during traffic monitoring in real-time detection, but testing phase is. The fastest algorithm in testing is Logistic Regression, but it is not useful because the average precision is zero in the previous experiment. The next fastest algorithm is Decision Tree by 0.00228sec. This algorithm was found to be more than 90% in classification evaluation.

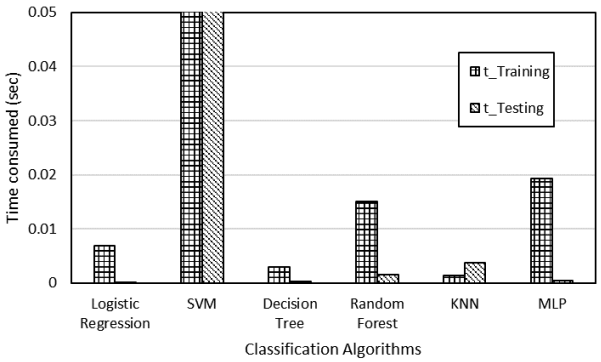


Fig. 4. Training Time and testing time of each algorithm.

E. Discussion

As a result, Random Forrest had the best performance in Precision, and Decision Tree had the best performance in time consumed. Considering two factors, Decision Tree is better. The Random Forest has achieved a higher degree of accuracy, but those are a slight difference. Since the decision tree is faster, it will be more advantageous for real-time detection. Another reason why decision trees are better is that it is white-box based learning. We can see the generated tree which features are more important and can use for feature selection

F. Weakness

Because proposed method is a passive, it cannot be used if packets cannot be collected. The detector sniffs and classifies packets at the boundary of the network. However, it will be useful to find a user who uses Freenet in the internal network. Underfitting can occur due to untrained applications. The type of flow data collected in this experiment is from the applications frequently used UDP, but those are based on experience. In other words, if some application has a flow similar to that of Freenet, a False Positive may occur. When the host behind the NAT sends the packet to the outside, the IP address field of the packet header is mapped to the address of the NAT device. Detection can be difficult if one host is running multiple applications that use UDP.

V. CONCLUSION

In this paper, we proposed classification of Freenet Traffic flow using Machine Learning techniques. This method performs a classification algorithm after filtering flows that do not match the Freenet characteristic among the collected flows. Through this, it is possible to identify the host operating the Freenet inside the network without joining Freenet. We also evaluated the performance of classification algorithms. Among them, Decision Tree was most effective with 94% of precision and 0.0029 sec of time spent.

Currently, we are working on building a real-time detection system using this classification algorithm. It is

considered to add new features and validation metrics to improve accuracy. As a future work, we will improve the algorithm to achieve a similar effect with fewer packet monitoring as a future work.

ACKNOWLEDGMENT

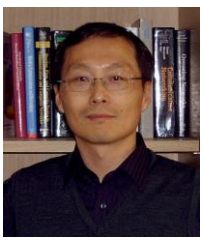
This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIP) (no. NRF-2015R1A2A2A01005577)

REFERENCES

[1] I. Clarke, O. Sandberg, B. Wiley, and T. Hong, "Freenet: A distributed anonymous information storage and retrieval system," in *Proc. Intl. Wkshp. Designing Privacy Enhancing Technologies*, pp. 46–66, 2001.
[2] Tor. [Online]. Available: https://www.torproject.org/
[3] I2P. [Online]. Available: https://geti2p.net/en/
[4] Facebook over Tor. [Online]. Available: https://www.facebook.com/facebookcorewwwi/?hc_ref=PAGES_TIMELINE&fref=nf
[5] A. Lazarenko and S. Avdoshin, "Anonymity of Tor: Myth and Reality," in *Proc. ACM the 12th Central and Eastern European Software Engineering Conference*, Nov. 2016
[6] Symentec Report: Ransom Wannacry. [Online]. Available:https://www.symantec.com/security_response/writeup.jsp?docid=2017-051310-3522-99
[7] T. Baumeister, Y. Dong, Z. Duan, and G. Tian, "A Routing Table Insertion (RTI) attack on freenet," in *Proc. Intl. Conf. on Cyber Security*, 2012.
[8] G. Tian, Z. Duan, T. Baumeister, and Y. Dong, "A traceback attack on freenet," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1797–1805.
[9] G. Tian, Z. Duan, T. Baumeister, and Y. Dong, "Thwarting traceback attack on freenet," in *Proc. IEEE GLOBECOM*, Dec. 2013, pp. 741–746.
[10] G. Tian, Z. Duan, T. Baumeister, and Y. Dong, "Reroute on loop in anonymous peer-to-peer content sharing networks," in *Proc. IEEE Conf. Communications and Network Security*, pp. 409–417, Oct. 2014.
[11] S. Roos, F. Platzer, J. Heller, and T. Strufe, "Inferring obfuscated values in Freenet," in *Proc. NetSys*, Mar 2015, pp. 1–8.
[12] B. Levine, M. Liberatore, B. Lynn, and M. Wright, "Statistical detection of downloaders in Freenet," in *Proc. IEEE International Workshop on Privacy Engineering*, May 2017.
[13] D. Kleinbaum and M. Klein, "Analysis of matched data using logistic regression," in *Logistic Regression*, Springer New York, 2010, pp. 389-428.
[14] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annual ACM Workshop on Computational Learning Theory*, 1992, pp. 144–152.
[15] J. Quinlan, "Induction of decision trees," *Machine Learning*, vol. no. 1, pp. 81-106, 1986.
[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no.1, pp. 5-32, 2001.
[17] E. Fix and J. Hodges, "Discriminatory analysis nonparametric discrimination: Consistency properties," Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
[18] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representation by error propagation," in *Parallel Distributed Processing*, vol. 1, Cambridge, MA: M.I.T. Press, 1986.
[19] J. Cai, Z. Zhang, and X. Song, "An analysis of UDP traffic classification," in *Proc. 12th IEEE Communication Technology*, Nov. 2010, pp. 116-119.

Seungwoon Lee received the B.S. degree in information and computer engineering and M.S. degrees in Software from Ajou University, Suwon, South Korea, in 2015 and 2017, respectively. He is currently working toward the Ph.D. degree at the Department of Computer Engineering, Graduate School, Ajou University, Suwon, South Korea. His research interests include network security and cyber warfare, future Internet networks, and military communications.

Seung-hun Shin received the B.S. degree in information and computer engineering and M.S. and Ph.D. degrees in information and communication engineering from Ajou University, Suwon, South Korea, in 2000, 2002, and 2011, respectively. From September 2011 to February 2016, he was with the Department of Software Convergence Technology, Ajou University as a Lecture Professor. Since March 2016, he has been with the Da-san University College, Ajou University, as an Assistant Professor. His research interests include software testing algorithms, network intrusion detection, and mobile multimedia networking.



Byeong-hee Roh received the B.S. degree in electronics engineering from Hanyang University, Seoul, South Korea, in 1987, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 1989 and 1998, respectively. From 1989 to 1994, he was with Telecommunication Networks Laboratory,

Korea Telecom, as a Researcher. From February 1998 to March 2000, he worked with Samsung Electronics Co., Ltd., South Korea, as a Senior Engineer. Since March 2000, he has been with the Department of Software and Computer Engineering and the Department of Computer Engineering, Graduate School, Ajou University, Suwon, South Korea, where he is currently a Professor. During 2005, he was a Visiting Associate Professor with the Department of Computer Science, State University of New York, Stony Brook, NY, USA. During 2014, he was an Adjunct Researcher with the Agency for Defense Development, South Korea. His research interests include mobile multimedia networking, future Internet networks, Internet of Things platform and services, network security and cyber warfare, and military communications.