Automatic Payload Signature Update System for the Classification of Dynamically Changing Internet Applications

Kyu-Seok Shim[1], Young-Hoon Goo[1], Dongcheul Lee[2]* and Myung-Sup Kim[1]*
[1]Dept. of Computer and Information Science
Korea University, Korea
[2]Dept. of Multimedia Engineering
Hannam University, Korea
[e-mail: {kusuk007, gyh0808, tmskim}@korea.ac.kr[1], jackdclee@hannam.ac.kr[2]]
*Corresponding authors: Dong-Cheul Lee and Myung-Sup Kim

Received August 4, 2017; revised March 18, 2018; revised June 17, 2018; accepted July 12, 2018; published March 31 2019

Abstract

The network environment is presently becoming very increased. Accordingly, the study of traffic classification for network management is becoming difficult. Automatic signature extraction system is a hot topic in the field of traffic classification research. However, existing automatic payload signature generation systems suffer problems such as semi-automatic system, generating of disposable signatures, generating of false-positive signatures and signatures are not kept up to date. Therefore, we provide a fully automatic signature update system that automatically performs all the processes, such as traffic collection, signature generation, signature management and signature verification. The step of traffic collection automatically collects ground-truth traffic through the traffic measurement agent (TMA) and traffic management server (TMS). The step of signature management removes unnecessary signatures. The step of signature generation generates new signatures. Finally, the step of signature verification removes the false-positive signatures. The proposed system can solve the problems of existing systems. The result of this system to a campus network showed that, in the case of four applications, high recall values and low false-positive rates can be maintained.

Keywords: Automatic, Traffic Classification, Association Rule Mining, Payload Signature

This work was supported by Institute for Information & communications Techonology Promotion(IITP) grant funded by the Korea Government(MSIT) (No.2018-0-00539-001,Development of Blockchain Transaction Monitoring and Analysis Technology) and by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No. 2017-0-00513, Developing threat analysis and response technology based on Security Analytics for Heterogeneous security solution)

http://doi.org/10.3837/tiis.2019.03.009 **ISSN : 1976-7277**

1. Introduction

In the network management field, traffic analysis is becoming increasingly important. Network traffic has a wide variety of forms as a result of the expansion of the network environment. Optimally efficient usage of network resources and the provision of smooth services to network users are the goals of network management. For achieving these goals, the most important element is network monitoring. Network monitoring consists of determining the traffic amount of a specific application and establishing a management policy that is suitable for it. In network monitoring, traffic classification is essential for providing a high quality of service to a user and ensuring that a high quality of service is received from a network provider, while minimizing network resources required. A signature is an essential feature in the traffic classification precess. It is used to classify traffic through the application that generates it. Obviously, a wide range of signatures exist, which can be classified according to the characteristics of the traffic[1,2].

In this paper, we focus on payload signatures in various levels. The payload signature is a unique and continuous substring within the traffic payload generated by the same application. However, to generate the payload signature, a considerable amount of time and money is required. Most existing methods use similar means to generate the payload signature. First, a manager gathers the traffic of an application to extract the signature, and then, the user finds the substring that commonly occurs by comparing the contents of the payloads. After its extraction the common substring can uniquely be used to verify an application. Depending on the extraction operator, there can be a difference in the quality of the signature, which leads to disadvantages in terms of signature objectivity. Finally, since the signature extraction operation consumes time and necessitates frequent operations, it is either difficult or impossible to update the signatures of all the applications.

To solve these problems, the studies of automatic payload signature generation have been conducted[3-8]. Most of these studies used a method of automatically extracting common substrings from payload in packets. However, these studies suffered limitations. First, when generating the signatures, the user must collect the application traffic directly. Second, disposable signatures can be generated because short-term traffic is collected and used. The signature set must be able to detect all functions of the application and it cannot be assumed that a signature extracted in a short time period can achieve this. Third, without the step of verification, some of the extracted signatures may be false-positive signatures. Finally, the signatures that are kept cannot always be the most recent ones. Because traffic patterns cannot be perceived to change, it is difficult to respond immediately unless the system create an environment that can be done in real-time.

We propose a automated signature update system that overcomes these limitations. The proposed system consists of the steps of traffic collection, signature management, generation, and verification. The limitations of the existing system can be overcome through each step. It compares the traffic data with the log data that are collected from the traffic measurement agent (TMA) for automatically collected ground-truth traffic[14]. Through this method, it is possible to extract only continuous signatures because temporary signatures are screened by the management process. A continuous signature can continuously detect traffic whereas a temporary signature can only temporarily detect traffic. As low accuracy signatures are removed in the verification process, only those signatures that have high accuracy are extracted. Finally, because the proposed system operates in real time, it is possible to response

immediately to changes in traffic patterns. In Section 2, we review the related work in the areas of automatic signature generation and traffic classification. In Section 3, we propose a automatic signature update system. We evaluate the proposed system in Section 4 and finally present the conclusion and future work in Section 5.

2. Related Work

As mentioned in Section 1, signatures for traffic classification are presented in a variety of forms according to the traffic characteristics. First, signatures based on port numbers analyze the traffic by using the port number provided by the Internet Assigned Numbers Authority (IANA). Second, signatures based on statistical information analyze the traffic by using statistical information, such as the size, location, and time of traffic. Signatures based on the payload analyze the traffic by using the payload of a packet in the data part. **Table 1** shows the properties and disadvantages of the signatures according to their type.

Table 1. Property and disadvantages of types of signature

Type	Example	Properties	Disadvantages
Port [9]	80: HTTP 21: FTP	Uses a fixed port number	Lack of accuracy
Payload [10]	“GET” “HOST”	Uses a unique pattern in the payload	Difficulty generating signatures
Statistic [11]	Packet size [+800, -1500, +200]	Uses statistical information	-Difficulty generating signatures -Lack of accuracy
Behavior [17]	# of port, # of IP	Uses behavior patterns	Difficulty generating signatures

Signatures based on port numbers analyze traffic at high speed using a relatively small amount of memory. However, many applications use random port numbers for passing the firewall and IPS equipment, and therefore signatures based on port numbers are meaningless. A signature based on statistical information is a fast method for analyzing encrypted traffic. However, this method has difficulty generating signatures, is limit for some applications and suffers a lack of accuracy.

The payload signature can classify the traffic most accurately. Therefore, in this study we address signatures based on the payload. The payload signature is a common substring in the payload of a packet. Although this signature is the most accurate, its extraction process is difficult. The proposed automatic payload signature generation method solves these problems. It uses various algorithms such as the longest common string(LCS) algorithm, the Smith-Waterman algorithm, and Autosig. The most recent research on automatic payload signature generation is a study in which a sequential pattern mining algorithm was used[12-13].

LCS-based application signature extraction(LASER) is a typical method for modifying the LCS algorithm that extracts the signature in application traffic[3]. This method finds a common sequence string by using backtracking matrix and comparing two strings. This comparision in the extraction process is very time consuming.

The Smith-Waterman algorithm was originally proposed for the purpose of determining the degree of similarity of the deoxyribonucleic acid (DNA)[4]. It has, however, been used as an

automatic signature generation method. The method can find the set of common substrings. However, its disadvantages are similar to those of the LASER method.

The Autosig system can resolve the disadvantages mentioned above[19]. It requires pre-processing in the order of the traffic flow and grouping for adopting actual traffic by comparing two strings and post-processing for integrating the resulting substrings in a rule. The AprioriAll algorithm selects only the likely common substring that can be a signature. The candidate strings are increased from length-1 to length-k. There is no need for pre-processing and post-processing, and the time consumed is not excessive. Thus, in this system, a signature is extracted by using the AprioriAll algorithm in the signature generation step.

Existing methods still have shared limitations. First, the user should collect the traffic of each application manually. Second, the extracted signatures can include disposable signatures. Third, since no verification process is performed, the extraction of the signatures is certain to involve high false-positive rates. Finally, that the most recent signature is kept is a disadvantage. In this paper, we propose a fully automatic signature generation system that includes the collection of traffic, signature management, signature generation, and signature verification.

3. Fully Automatic Payload Signature Update System

In this section, our proposed fully automatic payload signature update system is described. The proposed system automatically collects traffic and generates, manages and verifies signatures. We also describe the mechanism of each process in every step. **Fig. 1**, shows the process of the fully automatic payload signature update system.

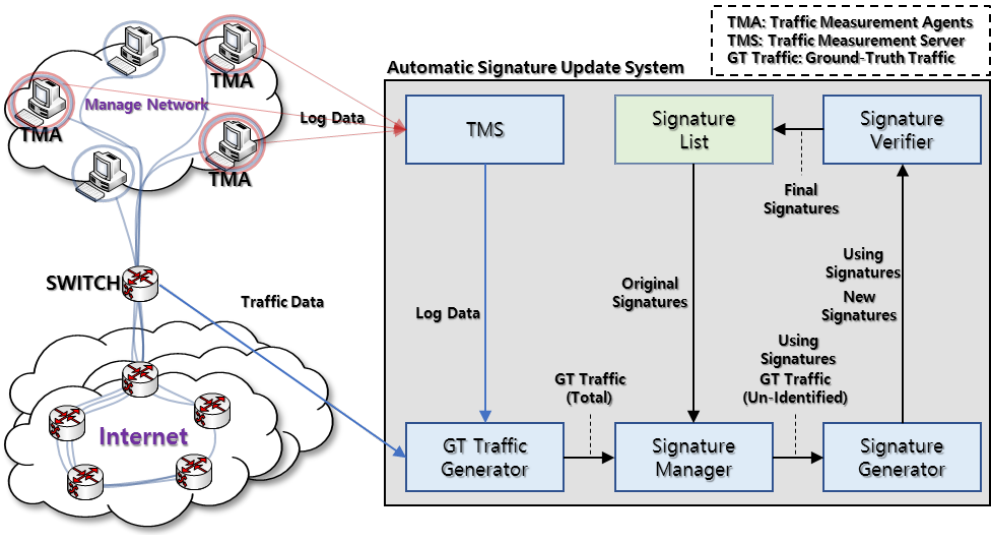

Fig. 1. Process of fully automatic payload signature update system.

GT Traffic Generator generates the ground-truth traffic using the traffic data and log data. The ground-truth traffic means the pure traffic of a particular application. GT Traffic Generator can collect traffic from each application as the log data containing the process name. Signature Manager separates the unanlyzed traffic by matching the entire ground-truth traffic with the existing signatures. In this step, unused signatures are deleted from the existing

signatures. If the unused signatures are deleted, the system can be overloaded. Signature Generator outputs new signatures entering un-identified traffics. Therefore, Signature Manager is the role to manage existing signatures, and Signature Generator is the role to create new signatures. The new signatures with a high false-positive rate are deleted through Signature Verifier.

3.1 GT Traffic Generator: Automatically Collecting the Ground-Truth Traffic

One major limitation of the methods developed in current studies traffic is collected manually. Although the GT traffic generator automatically collects the ground-truth traffic from each application. Most existing systems may collect inaccurate traffic at this stage. The proposed system uses the TMA for generating the ground-truth traffic[18]. The TMA is a program that periodically collects socket information of processes running on its host and provides it to specified server(TMS) with process path information. The TMA is installed on each host leaving the log data. **Table 2** shows the information included in the log data from the TMA.

Table 2. TMA information

Process name
IP address (local, remote)
Port number (local, remote)
State (start, continue, end, server)
Protocol
Path

The TMA sends the information shown in **Table 2** to the TMS at each host every 1 min. Then, the information obtained from the TMA of each host is integrated by the TMS. The GT traffic generator matches the traffic data and the TMA log data. This generates the ground-truth traffic for each application by matching at the time and a 5-tuple(srcIP/Port, dstIP/Port, Protocol). Twenty-four stored traffic files are collected per 1 hour. In this paper, we performed experiments on the four most frequently used applications: AfreecaTV (a video streaming service), uTorrent (a file sharing service), Kakaotalk (a messenger service) and Facebook (social network service).

3.2 Signature Management: Traffic Classification and Signature Management

The signature is generated using the collected ground-truth traffic. However, this method leads to a high system overload and is very time consuming as it consistently generates the same signatures of a same application. Furthermore, the management method must be applied to the unused signatures that exist among the signatures. The signature management stage manages the signatures by classifying unidentified traffic and deleting unused signatures.

Fig. 2 shows the process of unidentified traffic classification and signature management. In **Fig. 3**, “F” represents a flow. In the case of application X, flows 1, 5, and 7 are classified according to signatures 1 and 2. This method decreases the system overload as compared to the existing methods, as it does not consistently generate signatures. We need not only to generate new signatures but also to delete unused signatures for maintaining recent signatures. If this process is omitted, signatures will accrue and cause system overload, which results in excessive time consumption in the traffic analysis. Therefore, the proposed system deletes unused signatures. Equation 1 shows the configuration of a signature.

The signature consists of the header information, the content, and the score. The header information consists of source IP address, destination IP address, source port, destination port and L4 protocol. The content is the substring of applicaion's payload. Score is used to remove disposable signatures. Score indicates the number used in analysis. Score has an initial value of 1 with the signature generation. The signature was not used in the analysis are reduced by 1 point, and the signature used in the analysis are increased by 1 point. If the score is 0, the signature is removed. The reason for using the score in this system is to maintain the normal signatures and to remove the disposable signatures.

There are two cases in terms of the signatures being removed.The first case is a disposable signature generated by traffic that occurs only at a specific time. For example, disposable signatures include date and time keywords. The second case is a signature that is normal but does not occur in input traffic data. We use points because these signatures should not deleted. Finally, at this stage, disposable signatures are removed and unanalyzed traffic is classified.

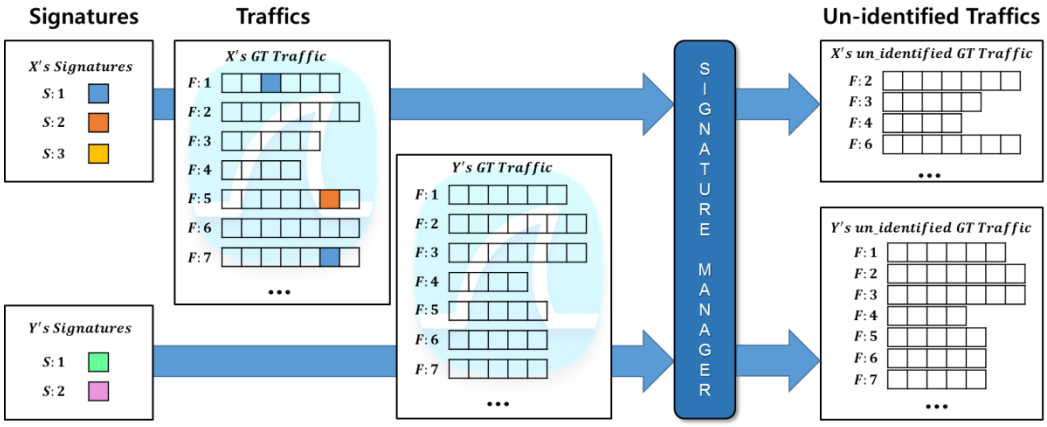

Fig. 2. Signature management process

$$Signature = \{Header, Contents, Score\} \quad (1)$$

$$Header = \{\, Destination\ IP\ address, Destination\ Port, L4Protocol\} \quad (2)$$

$$Contents = \{\, Set\ of\ common\ sub-string\,\} \quad (3)$$

3.3 Signature Generator: Automatic Signature Generation

The system automatically extracts signatures using a sequential pattern algorithm (AprioriAll)[15]. In this process, the system extracts the payload of the traffic and generates the sequences. In the generated sequences, the content signatures of length 1 consisting of alphabetic characters is generated. From the extracted length-1 content signatures unwanted content parts are deleted and thus length-2 content signatures are created. The process continues to length-k, when no more content signatures of a longer lengths can be generated from the extracted common strings. There are three types of signatures as shown in Fig. 3.

First, the content signature is a continuous and common string. Second, The second type is the packet signature. The packet signature is generated by a combination of content signatures from the same packet. The third type is the flow signature. The flow signature is generated by a combination of packet signatures from the same flow. In the step of content signature extraction, minimum support is given to a set of sequences and the sequences with a value higher than the minimum support is extracted. Thus, the output content set is a continuous strings that are frequently generated in the entered sequences.

This signature extraction method with Apriori algorithm increases lengh-k while removing content that is not satisfied with minimum support. This process is repeated until the length is no longer increased. The increased length of the content is completed by checking the inclusion relationship of the the sub-contents that completed the content and deleting the sub-contents when they are fully involved. Finally, the set of generated content sequences is passed to the subsequent packet signature extraction stage. This step is very similar to the content signature extraction stage. The content signature extraction process proceeds with the payload of packets in a conetnt sequence, but the packet signature extraction process proceeds with a packet sequence consisting of a combination of content signatures. **Fig. 4** shows the packet sequences extraction stage. As **Fig. 5** shows packet sequences are composed and packet signatures are extracted. The packet signature extraction stage increases the signatures' length from length-1 to length-k, which is a process similar to the content signature extraction process. In the packet signature extraction stage, length-1 candidates are the extracted content signatures. The packet signature is a combination of the content signatures existing in the same packet. **Fig. 6** shows the packet signature extraction process.

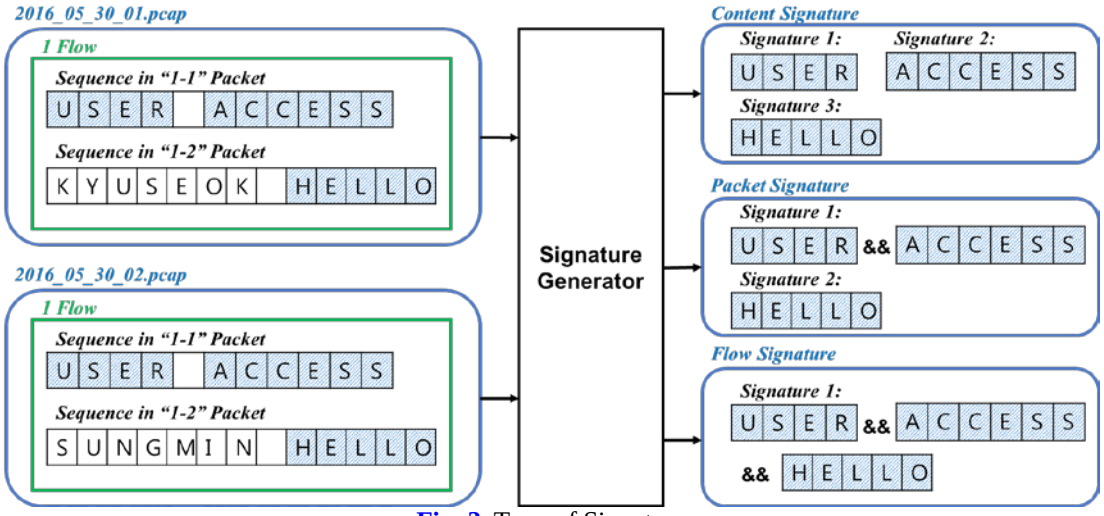

Fig. 3. Type of Signature

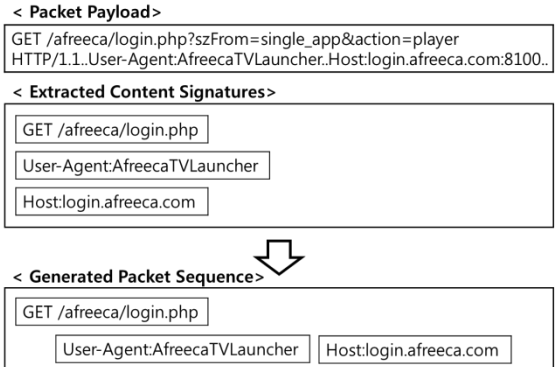

Fig. 4. Packet sequence extraction

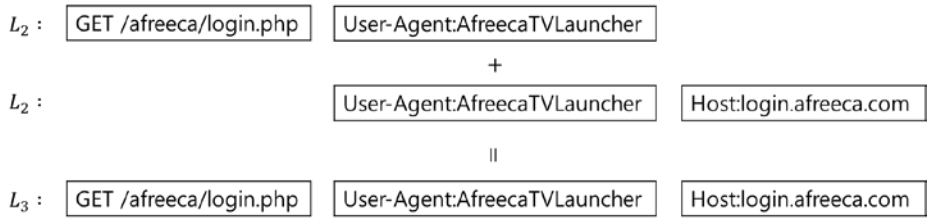

Fig. 5. Packet Signature extraction

Similarly to the packet signature extraction process shown in **Fig. 5**, content signatures occurring in the same packets are extracted and the process iterates until no more extraction is possible. When the packet signatures are extracted the inclusion relation is checked, in which step the included content signature subsets are deleted. Finally, the extracted packet signature set is passed on to the flow level signature extraction stage.

In the flow and packet signature extraction process, if the content signatures were used for packet signature composition, the flow signature uses packet sequences to compose their sequences. Therefore, the flow signatures refer to the packet signatures existing in the same flow. **Fig. 6** shows the extraction of the flow sequences. This Fig. also shows the flow sequences composition and flow signatures extraction processes. From the previous stage, the flow signatures are generated by using the extracted packet set signatures occurring in the same flow. The process iterates until it obtains the longest length-k signature, at which point no more possible longer length signatures can be generated. After the flow signatures are generated, the inclusion relation is checked and the packet signatures are deleted. Finally, in the flow signature set, the extracted content, packet and flow signatures are retained after the subset deletion process is completed.

As indicated below, traffic classifcation requries packet siganature and flow signature as well as content signature, a common strings. Because if traffic is classified just by content signatures, then coverage is high while the false-positive rate is also high. In comparison, packet signatures are more accurate than content signatures by increasing conditions because they consist of several content signatures. In this paper, it is not only packet signatures, but also flow signature is generated, generating flow signatures more accurate than packet signatures. Through this process, the false-positive rate is signigicantly reduced in signature generation.

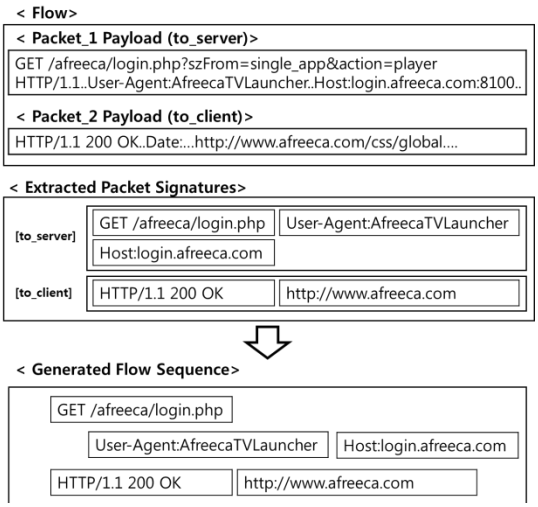

Fig. 6. Flow Sequence Extraction

3.4 Signature Verifier: Verification of Signatures

The proposed system includes a verification step for extracted signatures in the process. This system generates flow signatures with low false-positive rate. This step is essential, because a flow signature may include possible false-positives. The signature loses its meaning at the moment of classification of any application, not a specific application. The signature verification is aimed at signatures analyzing other applications that is, false-positive signatures. Signatures that are not false-positive are included in the final signature. In the verification process, the system will use the false-positive value. True-Positive(TP) is the value that analyzes and identifies an application A from a signature A. False-Negative(FN) is the value that does not analyze and identify an application A from a signature A. False-Positive(FP) is the value that analyzes and identifies an application from a signature A. True-Negative(TN) is the value that does not analyze and identify different applications from a signature A.

F-measure is a formula that can measure both the precision value and the recall value at the same. When using F-measure, users can determine the importance of precision and recall. The more weight add to the precision, the lower the β value is than 1, and the value of the F-measure depends on the precision value. The more weight add to the recall, the higher the β value is than 1, and the value of F-measure depends on the recall value. If two values are weighted equally, the β value will be 1. In this paper, a fixed value of 0.1 was used for β in F-measure. This is because the signature is accurate when the precision value is high. Equation 6 represents the expression of the F-measure. F-measure has a maximum of 1 and minimum of 0. In this paper, only signatures with F-measure values greater than or equal to 0.95 will be used as the end. Through the proposed method the signature is updated continuously, and an unused signature is removed and a new signature is extracted. A new signature maintains high accuracy during the verification process.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (4) \qquad Recall = \frac{TP}{TP + FN} \quad (5)$$

$$\text{F} - \text{measure} = \frac{(\beta^2 + 1) \times Precision \times Recall}{\beta^2 \times Precision + Recall} \quad (\beta = 0.1) \quad (6)$$

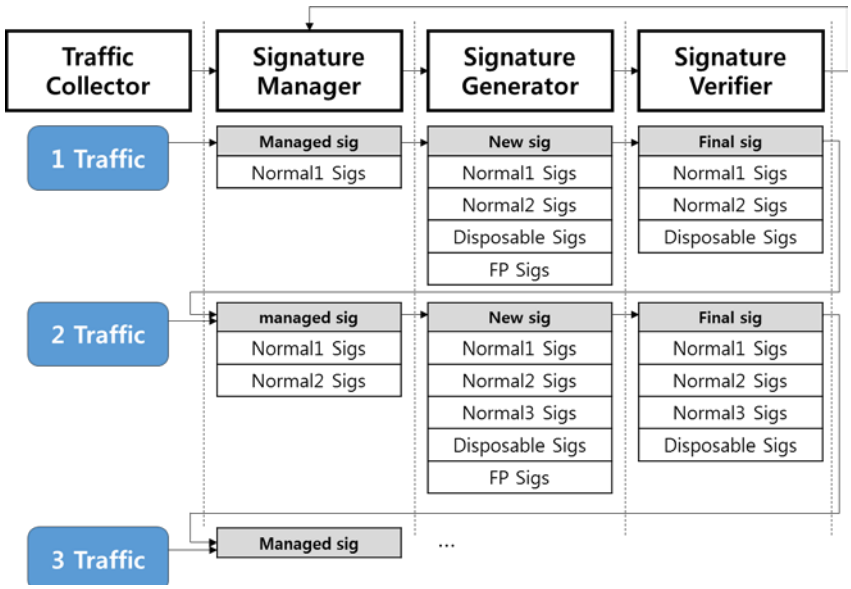

Fig. 7. The effects expected to be performed on a proposed system continuously

Fig. 7 illustrates the effects when the proposed system is continuously operated. If the automatic payload signature updates system is continuously operated on one application, it will accumulate only normal signatures, as shown in Fig. 8. By means of the signature manager, we use the cumulative score to remove disposable signatures. The signature generator also extracts three types of signatures: normal signatures that are not disposable and can be used to analyze only particular applications, disposable signatures that apply only to one-time traffic, and false-positive signatures. However, false-positive signatures are removed at the signature verification stage. In addition, disposable signatures are deleted at the signature management step in the next cycle.

4. Evaluation

In this section, we evaluate the performance of the fully automated payload signature generation system. We chose the four most frequently used applications for experimentation and for evaluation of the system using automatically collected traffic. We collected ground-truth traffic after installing the TMA on eight hosts using traffic from these eight hosts and TMA log data. The four applications were AfreecaTV to represent video and broadcast services, Facebook to represent social network services, Kakaotalk to represent messaging services, and uTorrent to represent file sharing and transfer services.

We grouped the collected traffic into three types: the initial traffic in order to extract the signatures of the year 2015, the updated traffic for the system cycle, and the evaluation traffic. The reason for using the 2015 traffic was to evaluate the importance of maintaining up-to-date signatures by analysis and comparison with the traffic of this year. Table 3 shows the information about the traffic collected in 2015.

Table 3. Traffic trace information in 2015

	AfreecaTV	Facebook	Kakaotalk	uTorrent
Date	2015.05.13	2015.03.24	2015.05.12	2015.05.12
Flow	1,813	279	444	694
Packet	148,743	55,986	218,923	930,803
Byte	120MB	51MB	334MB	850MB

Table 4. Traffic trace information for the signature update

	Update 1			Update 2		
	flow	pkt	byte	flow	pkt	byte
AfreecaTV	2,207	103,235	90MB	2,103	113,635	101MB
Facebook	239	88,254	77MB	210	326,254	307MB
Kakaotalk	11,612	419,452	306MB	11,250	355,451	245MB
uTorrent	154	45,623	48MB	67	21,368	22MB
	Update 3			Update 4		
	flow	pkt	byte	flow	pkt	byte
AfreecaTV	851	66,265	60MB	369	96,578	90MB
Facebook	221	329,784	309MB	260	291,968	281MB
Kakaotalk	12,532	1,239,445	1,081MB	10,455	587,457	514MB
uTorrent	42	57,547	58MB	192	60,656	63MB

Table 4 shows the traffic trace information for the signature updates. This traffic is generated by matching the log data of the TMA for four days in the ground-truth traffic generation stage. In addition, new signatures are generated at the signature generation stage by creating a set of combined signatures. Finally, we define the final signatures of the signature set after deleting the false-positive signatures in the signature verification stage. The verified final signatures are used as original input signatures in the next cycle of the system.

Table 5 shows the recall values and the false-positive rates of the signatures according to the respective application-specific system process. In **Table 5**, original signature No.1 is the signature extracted in 2015. The traffic used the first update traffic. The original signature as No. 1 of the using signatures is the new signature extracted using unidentified traffic in the signature management step. The system deletes the false-positive signatures and the remaining signatures are used for verification. The final signature No. 1 is the same as the original signature No. 2.

When using the signatures from the original signatures, the number of signatures is decreased, but the recall does not show a large difference. For example, there are 140 signatures in original signature No. 1 of AfreecaTV and 2 signatures in using signature No. 1 of AfreecaTV, but the recall is the same, 0.9%. This result was confirmed by the disposable signatures extracted in the signature generation stage. We confirmed the increase in the number of signatures and, the recall values and false-positive rates when extracting the new signatures. According to this result, the system extract not only the normal signatures but also the false-positive signatures. This proves that the verification process is essential for extracting the final signatures. Then, we confirmed that the false-positive rate showed a large decrease. For example, the false-positive rate of new signature No. 1 of Facebook is 31.2%. After removing the false-positive signatures, the false-positive rate of final signature No. 1 of Facebook is 0.16%: that is, the false-positive rate is decreased by 31.04%. Therefore, the signature verification stage deletes signatures that are possible false-positives.

Table 5. Recall value and false-positive rates of the signature according to the respective application-specific system process

(Flows %)

		#Sig	Recall				False-Positive			
			AfreecaTV	Facebook	Kakaotalk	Torrent	AfreecaTV	Facebook	Kakaotalk	Torrent
Original Signature	No.1	140	0.9	0.0	0.0	19	0.0	0.0	0.0	0.0
	No.2	56	20	25.3	33.7	36.7	0.0	0.16	0.0	0.0
	No.3	53	29.5	66.4	51.3	36.7	0.1	0.0	0.0	0.0
	No.4	44	37.8	78.3	88.3	36.7	0.0	0.05	0.08	0.0
Using Signature	No.1	2	0.9	0.0	0.0	19	0.0	0.0	0.0	0.0
	No.2	22	30	24.1	33.7	36.7	0.0	0.0	0.0	0.0
	No.3	28	27.5	65.4	51.3	36.7	0.0	0.0	0.0	0.0
	No.4	27	36.1	76.3	88.3	36.7	0.0	0.0	0.08	0.0
New Signature	No.1	59	31.1	95.6	50.1	36.7	0.1	31.2	0.99	0.0
	No.2	59	36.3	100	97.5	36.7	0.7	32.3	22.2	0.0
	No.3	49	42.8	85.8	88.3	36.7	1.4	0.39	0.01	0.0
	No.4	43	42.2	85.2	96.2	90.7	1.4	0.0	0.01	56.1
Final Signature	No.1	56	30	25.3	33.7	36.7	0.0	0.16	0.0	0.0
	No.2	53	29.5	66.4	51.3	36.7	0.1	0.0	0.0	0.0
	No.3	44	37.8	78.3	88.3	36.7	0.0	0.05	0.01	0.0
	No.4	40	36	85.2	96.2	41.1	0.0	0.0	0.01	0.0

Table 6. Recall values and false-positive rate when the F-measure is not used

	#sig	Recall			False-positive		
		flow	pkt	byte	flow	pkt	byte
New	28	96.2	76.1	72	0.08	0.01	0.00
Final	27	76.2	67.7	66.3	0.00	0.00	0.00

Table 7. Recall values and false-positive rates when the F-measure is used

	#sig	Recall			False-positive		
		flow	pkt	byte	flow	pkt	byte
New	28	96.2	76.1	72	0.08	0.01	0.00
Final	28	96.2	76.1	72	0.08	0.01	0.00

Autosig enters the collected ground-truth traffic in the signature generation module and extracts the new signatures. Therefore, it is the same as the proposed system without the deletion of unused and false-positive signatures. As compared to the proposed system, the Autosig system has a greater number of signatures and produces a greater number of false-positives. Moreover, Autosig cannot easily immediately update the signatures when the application traffic pattern changes because it conducts the operation only once without repeatition.

In the final experiment, the signatures were verified by using the F-measure in the signature verification stage[16]. We identified a significant difference according to whether an F-measure value was or was not used. The reason for using the F-measure value was to maintain the significant effect on the recall value without deleting small number of false-positive signatures. The experiment was conducted on Kakaotalk. **Tables 6** and **7** show the difference. **Table 6** shows the results of deleting all false-positive signatures, whereas. **Table 7** shows the results of maintaining a signature with a 20% recall value and with a 0.08% false-positive rate using the F-measure. New signatures significantly affect the recall value, but there is a possibility of false-positives. If the signature has a 0% false-positive rate without using the F-measure value, then a decrease in the recall value is seen. However, the results of this experiment confirmed that the recall value can be maintained by using the proposed F-measure.

5. Conclusion and Future Work

In this paper, we proposed an automatic system that performs ground-truth traffic collection, signature management, signature generation and signature verification. This system resolves the problems of the existing systems in which traffic is collected manually or semi-automatically. In addition, the limitation caused by the extraction of disposable signatures was solved by selecting only the signatures used in the signature management. The problem of the extraction of false-positive signatures was resolved in the signature verification stage. In this study, the proposed system accumulated only normal signatures and deleted disposable and false-positive signatures. Finally, the system increased the recall values and decreased the false-positive rates as the number of execution times increased.

In future work, we will adopt other methods of collecting ground-truth traffic that are different from using the TMA and TMS, because to allow the TMA method to classify ground-truth traffic, it is necessary to install a TMA to install to each host. We also plan to apply more optimized algorithms to improve the speed of the current system.

Reference

[1] M.-S. Kim, Y. J. Won, and J. W.-K. Hong, "Application-level traffic monitoring and an analysis on IP networks," *ETRI journal*, vol. 27, pp. 22-42, 2005. Article (CrossRef Link)

[2] B. Park, Y. Won, J. Chung, M. S. Kim, and J. W. K. Hong, "Fine-grained traffic classification based on functional separation," *International Journal of Network Management*, vol. 23, pp. 350-381, Sep 2013. Article (CrossRef Link)

[3] B.-C. Park, Y. J. Won, M.-S. Kim, and J. W. Hong, "Towards automated application signature generation for traffic identification," in *Proc. of Network Operations and Management Symposium, NOMS 2008*, IEEE, pp. 160-167, 2008. Article (CrossRef Link)

[4] X. Feng, X. Huang, X. Tian, and Y. Ma, "Automatic traffic signature extraction based on Smith-waterman algorithm for traffic classification," in *Proc. of Broadband Network and Multimedia Technology (IC-BNMT), 2010 3rd IEEE International Conference on*, pp. 154-158, 2010.

[5] H.-A. Kim and B. Karp, "Autograph: Toward Automated, Distributed Worm Signature Detection," in *Proc. of USENIX security symposium*, 2004.

[6] Y. Wang, Y. Xiang, and S. Z. Yu, "An automatic application signature construction system for unknown traffic," *Concurrency and Computation-Practice & Experience*, vol. 22, pp. 1927-1944, Sep 2010. Article (CrossRef Link)

[7] Y. Choi, An Automated Classifier Generation System for Application-Level Mobile Traffic Identification, 2011.

[8] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, "ACAS: automated construction of application signatures," in *Proc. of the 2005 ACM SIGCOMM workshop on Mining network data*, pp. 197-202, 2005. Article (CrossRef Link)

[9] IANA port number list. Available: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml

[10] T. Choi, C. Kim, S. Yoon, J. Park, B. Lee, H. Kim, et al., "Content-aware internet application traffic measurement and analysis," in *Proc. of Network Operations and Management Symposium, 2004. NOMS 2004, IEEE/IFIP*, pp. 511-524, 2004. Article (CrossRef Link)

[11] N. F. Huang, G. Y. Jai, H. C. Chao, Y. J. Tzang, and H. Y. Chang, "Application traffic classification at the early stage by characterizing application rounds," *Information Sciences*, vol. 232, pp. 130-142, May 2013. Article (CrossRef Link)

[12] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. of Data Engineering the Eleventh International Conference on*, pp. 3-14, 1995. Article (CrossRef Link)

[13] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. of 20th int. conf. very large data bases*, VLDB, pp. 487-499, 1994.

[14] S.H.Yoon, H.G.No, M.S.Kim, "Internet Application Traffic Classification using the TMA(Traffic Measurement Agent)," in *Proc. of 29th KIPS*, Daegu, KyungIl University, Vol.15, No.1, pp.946-949, May. 17, 2008.

[15] K.S.Shim, S.H.Yoon, S.K.Lee, S.M.Kim, W.S.Jung, M.S.Kim, “Automatic Generation of Snort Content Rule for Network Traffic Analysis,” *KICS*, Vol.40, No.04, pp.666-677, April, 2015. Article (CrossRef Link)

[16] D.M.POWERS, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, Dec, 2011.

[17] S.H. Yoon, J.S. Park, and M.S. Kim , "Behavior Signature for Fine-grained Traffic Identification," *Applied Mathematics & Information Sciences*, Vol. 9, No. 2L, pp. 523-534, Apr. 2015.

[18] S. H. Yoon, H. G. Roh, and M. S. Kim, “Internet Application Traffic Classification using Traffic Measurement Agent” *KIPS Commun.*, pp. 946-949, Daegu, Korea, May 2008.

[19] M. Ye, K. Xu, J. Wu, and H. Po, “Autosig-automatically generating signatures for applications,” in *Proc. of the 9th IEEE International Conference on Computer and Information Technology*, pp. 104-109, 2009. Article (CrossRef Link)

Kyu-Seok Shim received his B.S degree in Computer Science from Korea University, Korea, in 2014. He is currently a Ph.D candidate student of Korea University, Korea. His research interests include Internet traffic classification and network management.

Young-Hoon Goo received his B.S. degree in Computer Science from Korea University, Korea, in 2016. He is currently a Ph.D candidate student(Integrated Program) of Korea University, Korea. His research interests include Internet Traffic classification and network management.

Dongcheul Lee has been an assistant professor in the Department of Multimedia at Hannam University, Korea since 2012. He received the B.S. and M.S. degrees in Computer Science and Engineering from POSTECH, Korea in 2002 and 2004, respectively, and Ph.D. degrees in Electronics and Computer Engineering from Hanyang University, Korea in 2012. He had been a senior researcher at the KT Mobile Network Laboratory, Korea from 2004 to 2012. His main topics of interests have been communication in mobile applications, rich communication service, software framework for mobile games and machine learning in games.

Myung-Sup Kim received his B.S., M.S., and Ph.D. degree in Computer Science and Engineering from POSTECH, Korea, in 1998, 2000, and 2004, respectively. From September 2004 to August 2006, he was a postdoctoral fellow in the Department of Electrical and Computer Engineering, University of Toronto, Canada. He joined Korea University, Korea, in 2006, where he is working currently as an associate professor in the Department of Computer and Information Science. His research interests include Internet traffic monitoring and analysis, service and network management, and Internet security.