Regular Paper

Detection of Malware Infection based on the Similarity between Malware Infected Traffic

Masatsugu Ichino†, Yuuki Mori†, Mitsuhiro Hatada‡, and Hiroshi Yoshiura†

†Graduate School of Informatics and Engineering, The University of Electro-Communications, Japan
‡NTT Communications, Japan
ichino@inf.uec.ac.jp, y-mori@uec.ac.jp, m.hatada@ntt.com, yoshiura@uec.ac.jp

Abstract - New types of malware are appearing every day, and malware attacks have become an urgent problem. Current methods of detecting malware use malware signatures, which need to be identified and registered in advance. However, the daily appearance of new types of malware makes such identification and registration impractical. A more practical approach is to identify malware on the basis of traffic behavior since each malware type displays a unique behavior. We have developed a method for detecting malware infection using traffic models based on the similarity between traffic of malware samples. Malware-infected traffic is divided into clusters on the basis of traffic behavior, and a model representing each cluster is created. These models are used to identify target traffic samples as infected or normal. This method should enable the detection of infection caused by a new type of malware if the malware's traffic behavior is similar to that represented by one of the models. Simulation evaluation demonstrated that the proposed method can effectively identify malware-infect traffic with high accuracy. And we discussed the created models and effectiveness using the models created by proposed method. We also discussed the detection of unknown malware using the models created by proposed method.

Keywords: security, malware, malware detection, traffic, clustering

1 INTRODUCTION

New types of malware are appearing every day, and malware attacks have become an urgent problem. Current methods of detecting malware use malware signatures, which need to be identified and registered in advance. However, the daily appearance of new types of malware makes such identification and registration impractical. A more practical approach is to identify malware on the basis of traffic behavior.

This paper focuses on infection detection, which we broadly classify as malware detection, intrusion detection, and infection detection. Intrusion detection typically involve techniques for detecting unauthorized access from a network before a malware infection occurs. Infection detection involves techniques for detecting an existing malware infection from network traffic as usual. Malware infections have become more difficult for users to detect, so infections have spread more widely without users knowing that their computers are being used maliciously. Therefore, infection detection for personal computers and middleboxes in the network such as routers and firewalls is an important measure for preventing the spread of infection.

The research reported here focused on the use of traffic data to detect infection. This approach determines the features of normal communication traffic and of infected traffic and uses pattern recognition techniques to detect infections. Infection detection based on traffic data uses only the incoming and outgoing communication traffic of the target machine. Basically, traffic is generated if there is an infection, so this method holds promise as a means of detection from outside the target machine. That is, malware infections are externally detected by observing the target machine's traffic patterns when it connects to a network.

Each malware type displays a unique behavior and a unique communication pattern when an infected terminal is connected to a network. The unique communication pattern comprises association confirmation of the infected terminal to the internet, surveying of the network environment, communication between the command and control (C&C) server and the infected terminal, and so on. Whether the malware is known or unknown, some malwares exhibit common communication behavior when the terminal is infected with malware.

In this paper, we propose malware infection detection using the traffic models based on the similarity between malware-infected traffic samples. It works by creating feature values on the basis of the time series of the traffic data, clustering malware-infected traffic samples in accordance with the similarities between them, and creating a representative model for each malware cluster. Detection of unknown malware infection is important research theme. The target of this paper is to classify the malware infection traffic as malware infection. This paper focuses on the detection method of malware infection included in the unknown malware infection. Malware-infected traffic is divided into clusters on the basis of traffic behavior. A model of each cluster is created, and the models are used to identify target traffic as infected or normal. This method will enable infections caused by new types of malware to be detected if the resulting traffic behavior is similar to that represented by one of the models.

This paper is organized as follows. Section 2 introduces related work, Section 3 describes the proposed method, Section 4 describes the evaluation method, Section 5 presents the key results, Section 6 discusses the results, and Section 7 concludes the paper with a brief summary of the key points and a mention of future work.

2 RELATED WORK

There have been various studies of malware detection using traffic data. Some used the definitions provided by security vendors for detecting malware infection, and some did not.

Studies in the first group (e.g.,[1]-[5]) classified malware traffic samples into groups on the basis of the definitions and created models of infected traffic for each group. However, security vendor definitions are not always based on the characteristics of infected traffic. It is thus better to create models on the basis of the characteristics of infected traffic.

Some studies in the second group ([6]-[8]) created models of infected traffic on the basis of the characteristics of infected traffic but did not consider the time series of the traffic data and the similarities between malware-infected traffic samples. Other studies ([9]-[11]) created models of infected traffic on the basis of the characteristics of infected traffic considering similarities between malware-infected traffic samples but did not consider the time series of the traffic data. Still other studies ([12], [13]) created models of infected traffic on the basis of the characteristics of infected traffic considering the time series of the traffic data but did not consider the similarities between malware-infected traffic samples.

Traffic data is a stream of network information, and previous studies have demonstrated the effectiveness of considering the time series of traffic data. Consideration of the similarities between malware-infected traffic samples is also necessary for representing common characteristics of infected traffic.

Therefore, in our study, we created feature values by considering the time series of each malware traffic sample. Next, we divided the malware samples into clusters on the basis of their similarities. Then, we created models representing the common characteristics of the infected traffic for each cluster.

3 PROPOSED METHOD

Our method is based on the detection of malware-infected traffic by using models representing each common traffic characteristic of malware. Proposed method conducts three parts shown in Fig. 1. As outlined above, feature values are created by considering the time series of the traffic data, malware samples are clustered by considering the similarities between them, and a representative model is created for each malware cluster. Labels **(Step 1)** $\cdots$ **(Step 6)** used in the following subsection correspond to the step numbers in Section 4.2.1 , in which we describe the proposed method as an experimental procedure with experimental datasets.

3.1 Create Feature Values by Considering Time Series of Traffic Data

(Step 1) Extract features of training data

To create a feature vector representing the time series of the traffic data, the time series is divided into 1-s time slots, as shown in Fig. 2. The traffic data is a set of packets captured from network traffic. The time slots are then grouped into intervals lasting a defined number of seconds for analysis.

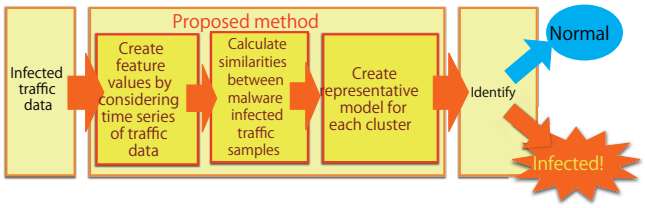


Figure 1: Outline of proposed method

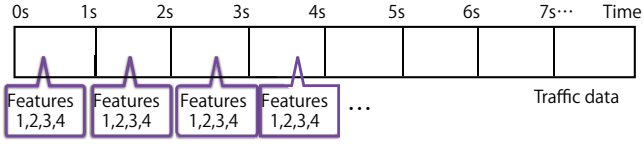


Figure 2: Division of time series of traffic data into 1-s time slots

Dividing time-varying traffic into time slots with a fixed duration and monitoring that traffic in units of time slots enables normal and infected traffic to be distinguished by focusing on the overall temporal variation in that traffic. In this work, we set the time-slot width to 1 s and determined the features for every time slot. The feature values are calculated for each time slot, and a feature vector concatenating the feature values is created for each time slot. In this study, we used minimum packet size per time slot, number of SYN packets per time slot, ratio of SYN packets to TCP packets per time slot, and number of ACK packets per time slot as the feature values. In our previous study, we analyzed traffic data after a malware infection and clarified which features would be the most effective in the detection of infection. It focused on using traffic data to detect infections and on the use of features that do not change much over time from those of the training data. In the evaluation, minimum packet size per time slot, number of SYN packets per time slot, ratio of SYN packets to TCP packets per time slot, and number of ACK packets per time slot as the feature values were effective features[9]. In this study, we represented the time-slot information as a four-dimensional feature vector by concatenating the feature values.

3.2 Cluster Malware Samples by Considering Similarities Between Malware Infected Traffic Samples

(Step 2) Create codebook for training data

To represent the traffic data as a code sequence, the set of feature vectors created as described in Section 3.1 is clustered (in this study, we used the LBG + splitting vector quantization algorithm [14]), and code is created for each cluster. A cluster is a mass of feature vectors and is divided on the basis of the distribution of data in the feature space. The code is the representative value of the cluster. The codebook is the code set. For example, when four-cluster clustering is applied, four codes (a, b, c, and d) are created. And we call the set of four codes the codebook.

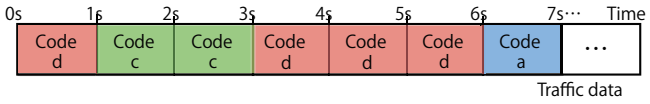


Figure 3: Example nearest code

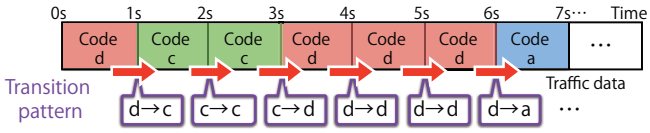


Figure 4: Example transition pattern

(Step 3) Create time-series representation of training data

The distances between the feature vector of target time slot and the code for each cluster is calculated, and a search is made for the nearest code. The time slot is then shifted, and a search is again made for the nearest code shown in Fig. 3. This series of nearest codes is called a "transition pattern." An example transition pattern is shown in Fig. 4.

The number of occurrences of each transition pattern per time interval (for example 40 s) is counted, and the ratio of each target transition pattern to all types of transition patterns is calculated, as shown in Fig. 6. The time interval is then shifted, and the number of occurrences of each transition pattern per time interval is again counted, and the ratio of each target transition pattern to all types of transition patterns is calculated.

(Step 4) Calculate similarity(correlation coefficient) between each pair of samples in training data

To evaluate the similarities between two malware traffic samples, their correlation coefficient is calculated using the occurrence frequency ratios, like those shown in Fig. 5. The correlation coefficient represents the correlation between each sample's digital sequence of the number of transition patterns $\times$ the number of time intervals. The coefficient is calculated using

$$\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2},} \quad (1)$$

where x and y are the probability variables, $\bar{x}$ is the mean value of x, $\bar{y}$ is the mean value of y, and n is number of transition patterns $\times$ the number of time intervals.

Calculation of the correlation coefficient requires that the n of x equals the n of y, as shown in Fig. 6. However, each malware traffic sample has a variable number of time intervals because each malware sample has a variable number of infected time intervals. The number of time intervals is thus adjusted by applying dynamic programming matching (DP matching) to the digital sequences of the two samples. The correlation coefficient is calculated using the adjusted digital sequences. DP matching adjusts the time lengths of the two samples by considering the time-series information and stretching the parts that are similar between the samples.

	a→a	a→b	a→c	a→d	b→a		d→c	d→d
0s~40s	0%	0%	0%	0%	0%	···	5%	82%
40s~80s	0%	0%	0%	5%	0%	···	0%	60%
···						···		

Figure 5: Example occurrence frequency ratios

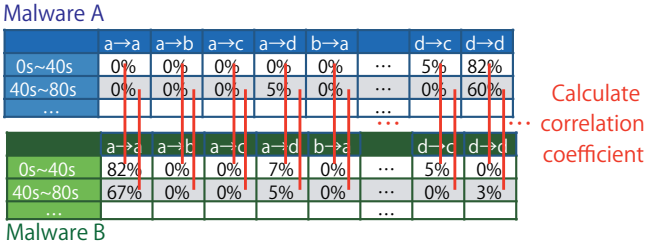


Figure 6: Calculate correlation coefficient

3.3 Create Representative Model for Each Malware Cluster

(Step 5) Create representative model for each malware cluster for training data

A representative model is created for each malware cluster by extracting a representative malware sample.

The malware samples are classified using hierarchical clustering based on correlation coefficients (we used the nearest neighbor method as hierarchical clustering). A high correlation coefficient means the similarity is high. The malware sample that has the most traffic samples with a correlation coefficient greater than an upper threshold is selected as the initial representative malware sample for the cluster. Since the optimal number of clusters is unknown in advance, hierarchical clustering is used as it does not require advance setting of the number of clusters. We extracted the malware sample that had the most malware's traffic samples with a correlation coefficient greater than the threshold(upper threshold) as the initial representative malware sample for the cluster. By the same token, the malware samples for which the correlation coefficient between the two malware traffic samples is less than the threshold (lower threshold) are deselected in each cluster to remove the samples for which the correlation is weak. This clustering is repeated until all training samples are divided into clusters.

The extracted malware traffic sample for a cluster is used as a representative model for that cluster in order to model sequential traffic data that actually occurred.

3.4 Detection of Infection

(Step 6) Calculate similarity between two samples in testing data

The time-series features of the testing data are created using Steps 1 and 3. The similarity between each representative cluster model and the target malware traffic sample is calculated, and the similarity between the model of normal traffic and the target malware traffic sample is calculated. The two similarities for each sample are compared, and the sample is

identified as infected or normal.

4 EVALUATION

4.1 ExperImental Datasets

We used the anti-Malware engineering WorkShop (MWS) Datasets [15] for our evaluation. In particular, we used the CCC (Cyber Clean Center) DATAset and the D3M (Drive-by Download Data by Marionette) Dataset for training. As malware-infected traffic data, we used 317 malware samples (151 from CCC DATAset 2010, 156 from CCC DATAset 2011, and 10 from D3M 2012) for the training data such that the data used for training were older than the data used for testing. The normal traffic data used for training were captured between 2011 and 2015.

We also used the CCC DATAset and D3M Dataset for testing. As malware-infected traffic data, we used 200 malware samples (177 from CCC DATAset 2011, 15 from D3M 2013, 5 from 2014, and 3 from D3M 2015) for the testing data such that the data used for testing were newer the data used for training.

The CCC 2010 and CCC 2011 attack data include communications prior to malware infection. Thus, given the purpose of our evaluation, we extracted from this attack traffic only the traffic following malware infection using the method described by our group et al. [9].

4.2 Experimental Methods

To evaluate the effectiveness of the proposed method, we compared its detection performance with that of three reference methods.

4.2.1 Detection Using Proposed Method

As shown in Fig. 7, using our proposed method, we performed six basic steps .

(Step 1) Extract features of training data

Step 1-1 We divided the traffic data into 1-s time slots. We used the packet header information because the payload information was often encrypted.

Step 1-2 From each time slot, we extracted four features that we had determined to be effective for infection detection: minimum packet size per time slot, number of SYN packets per time slot, ratio of SYN packets to TCP packets per time slot, and number of ACK packets per time slot. The four features are evaluated as effective features for infection detection in [9].

Step 1-3 We normalized the extracted feature values by using the min-max method.

Step 1-4 We represented the time-slot information as a four-dimension feature vector by concatenating the normalized values.

(Step 2) Create codebook for training data

Step 2-1 We applied the LBG+splitting vector quantization algorithm to the vectors with the cluster number set to four.

Step 2-2 We calculated a representative vector (cluster center) for each malware cluster and collected the vectors into a codebook representing the characteristics of each cluster.

(Step 3) Create time-series representation of training data

Step 3-1 We calculated the distances between the feature vector of time slot and each code. And we assigned code of minimum distance to the time slot. The time slot is then shifted, and a search is again made for the nearest code.

Step 3-2 We assigned a code to all time slots.

Step 3-3 We counted the frequency of each transition pattern in each time interval and represented the ratio of the frequencies as time-series information. There were 16 transition patterns (a→a, a→b, $\cdots$, d→c, d→d) because we used four codes. We set the time interval to 10, 20, 30, 40, or 60 s. For example, when we set the time interval to 10 s, we calculated the frequency of each transition pattern in each 10-s interval (comprising ten time slots) and calculated the ratio of the frequencies of each transition pattern. We then shifted the time interval and calculated the frequency of each transition pattern per interval and calculated the ratio of each target transition pattern to all types of transition patterns.

(Step 4) Calculate similarity (correlation coefficient) between each pair of samples in training data

We calculated the correlation coefficient between each pair of malware samples. A total of 50,086 ($=_{317} C_2$) correlation coefficients were calculated for each interval. We adjusted the time length (number of transition pattern $\times$ number of time interval) of each pair of malware samples by using DP matching.

(Step 5) Create representative model for each malware cluster for training data

We performed hierarchical clustering using the correlation coefficients calculated in Step 4. In multi-variant analysis, the correlation between each pair of samples was evaluated using the following criterion based on correlation coefficients C [16].

$0.0 \leq C \leq 0.2$: barely correlated

$0.2 \leq C \leq 0.4$: weakly correlated

$0.4 \leq C \leq 0.7$: a little strongly correlated

$0.7 \leq C \leq 1.0$: strongly correlated

The calculated correlation coefficients were used as measures of the similarity between malware-infected traffic samples. The higher the coefficient, the stronger the correlation. The malware samples for which the correlation was very high were grouped together. On the basis of the above criteria, a coefficient greater

than 0.7 generally means that the correlation is very strong. Therefore, we set the upper threshold to 0.7. The lower the coefficient, the weaker the correlation. The malware samples for which the correlation was very low were removed from the cluster. On the basis of the above criteria, a coefficient less than 0.2 generally means that the correlation is very weak. Therefore, we set the lower threshold to 0.2. Given these criteria, we selected the malware sample that had the most traffic samples with a correlation coefficient greater than 0.7 as the initial representative malware sample for the cluster. To keep a somewhat high correlation between each pair of malware traffic samples in the cluster, we deselected the malware samples that did not correspond to more than 70% of samples in the cluster; that is, the correlation coefficient was more than 0.2.

(Step 6) Calculate similarity between two samples in testing data

Step 6-1 We created the time series features of the testing data using Steps 1 and 3.

Step 6-2 We created a model of normal traffic using Steps 1 to 5.

Step 6-3 We calculated the cumulative minimum distance between each representative cluster model and the target malware traffic sample and calculated the cumulative minimum distance between the model of normal traffic and the target malware traffic sample.

Step 6-4 We compared the two distances for each sample. If the distance between the representative cluster model and the sample was greater than that between the model of normal traffic and the sample, the sample was identified as normal. Otherwise it was identified as infected.

4.2.2 Detection Using Time-Slot Method

For detection using time slots, we did not use both the time-series information and the similarity between malware-infected traffic samples. Instead, we created four codes from the malware infection training data using Steps 1 to 2 and created four codes from the normal training data using Steps 1 to 2.

We calculated the distances between the vector for the target time slot and the four codes for infection. Of the four distances calculated, the minimum one was selected as the similarity for infection. Moreover, the distances between the vector for the target time slot and the four codes for normal were calculated. Of the four distances calculated, the minimum distance was selected as the similarity for normal. Next, we compared the two similarities. If the one for infection was smaller than the one for normal, the time slot was identified as infected. If the one for normal was smaller than the one for infection, the time slot was identified as normal.

We applied the same process to all time slots of each malware traffic sample. If the ratio of the number of infected time slots to number of all time slots was more than the threshold (20%, 50%, or 70%), we identified the target traffic sample as malware-infected.

4.2.3 Detection Using One Representative Model

For detection using one representative model, we used the time-series information. We did not use the similarity between pairs of malware samples. The average malware traffic sample of the training data was treated as the representative model of malware-infected traffic.

We created the time-series information for the target malware traffic samples using steps 1 to 3 above. We calculated the mean ratio of the frequencies of each transition pattern for all malware traffic samples and selected the sample that was closest to the mean as the representative model of malware-infected traffic.

For testing, we created time-series information for the malware traffic samples using steps 1 to 3 above. We calculated the cumulative minimum distance between the target sample and the model of infected traffic. We also calculated the cumulative minimum distance between the sample and the model of normal traffic and identified the sample as normal or malware-infected on the basis of the two distances.

4.2.4 Detection Using Models Based on Security Vendor's Definitions

For detection using models based on a security vendor's definitions, we used the time-series information and clusters for classification. We did not use the similarity between malware-infected traffic samples.

We created time-series information for the target malware traffic sample using steps 1 to 3 above. Next, we divided the training malware traffic samples into clusters defined by the security vendor: BKDR, PE, Mal, TROJ, andWORM. We calculated the mean ratio of the frequencies of the transition patterns of the malware samples in each cluster. We selected the sample in each cluster with the frequency closest to the mean as the representative model of malware-infected traffic.

We calculated the cumulative minimum distance between the model of malware-infected traffic and target traffic sample and calculated the cumulative minimum distance between the model of normal traffic and target sample. We identified the sample as normal or infected on the basis of the distances.

5 RESULTS

5.1 Identification Rate of Proposed Method

The identification rate of the proposed method by changing the time interval is summarized in Table 1. The time interval is the duration during which the code transitions were counted, as described in section 3.2. The number of patterns of infected traffic is the number of hierarchical clusters, as described in section 3.3. The identification rate is the number of correctly identified malware-infected traffic samples divided by the total number of such samples in the testing data.

The identification rate was 100% for time intervals of 10, 20, 30, and 40 s, meaning that it is robust against the time interval.

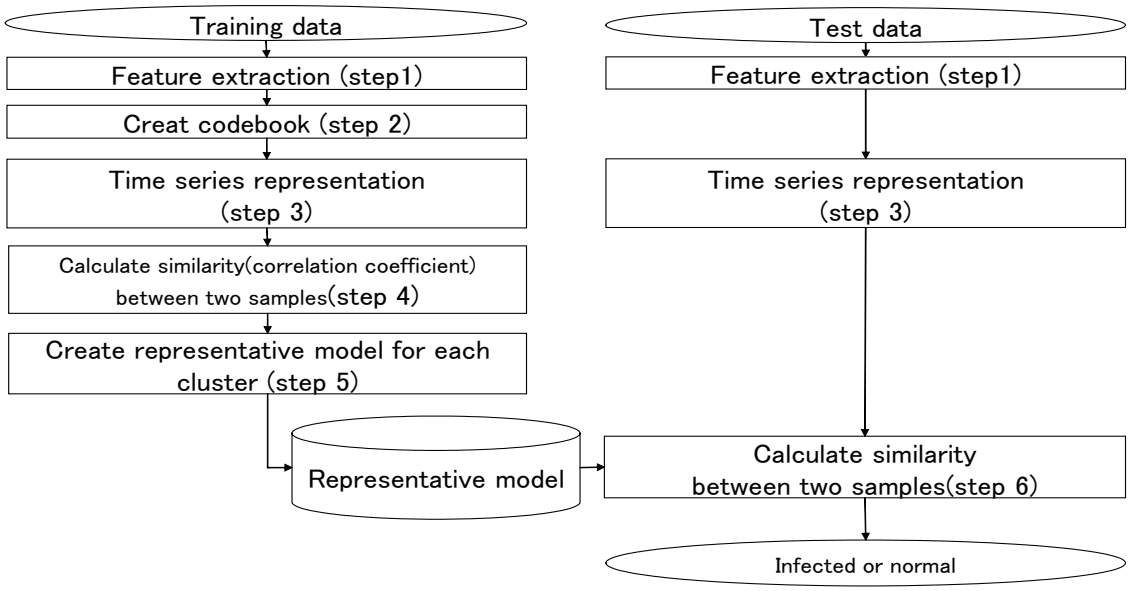


Figure 7: Overview of experiment

Table 1: Identification rate of proposed method

Time interval (s)	No. of patterns of infected traffic	Identification rate (%)
10	17	100
20	15	100
30	12	100
40	12	100
60	11	99.0

Table 2: Identification rate of time-slot method

Rate of infected time slots (threshold)	Identification rate
20%	87.0%
50%	22.5%
70%	16.5%

5.2 Identification Rate of Other Methods

To evaluate the effectiveness of proposed method, we compared its identification rate with those of the three reference methods. The identification rate of the time-slot method (section 4.2.2) is shown in Table 2. The rate of infected time slots is the number of time slots identified as infected divided by the total number of infected time slots in each traffic data. It is used for identifying whether a traffic sample is infected or normal. The identification rate of the one-representative-model method (section 4.2.3) is shown in Table 3. The identification rate of the security-vendor-definition-based method (section 4.2.4) is shown in Table 4. The number of patterns of infected traffic is five because the data used included five malware families.

Table 3: Identification rate of one-representative-model method

Time interval (s)	No. of patterns of infected traffic	Identification rate (%)
10	1	12.5
20	1	14.5
30	1	25.5
40	1	32.5
60	1	52.5

Table 4: Identification rate of security-vendor-definition-based method

Time interval (s)	No. of patterns of infected traffic	Identification rate (%)
10	5	47.5
20	5	78.0
30	5	92.0
40	5	87.5
60	5	98.5

6 DISCUSSION

6.1 Representative models

We first discuss the effectiveness of increasing the number of models, i.e., the number of patterns of infected traffic. As shown in Tables 1, 2, 3, 4, the proposed method had the highest identification rate.

To evaluate [the effectiveness to represent some models of infected traffic, we analyzed the main transition pattern of each model. As described in section 3.3, a transition pattern is the transition of the codes in a time slot. To analyze the main transition pattern of each model, we show the traffic features of each code by analyzing the traffic data near the code. The

Table 5: Traffic Features of Each Code

Code	Feature(s)
a	SYN send
b	ACK send
c	UPnP, CBrowsing, SSL communication (digital sign etc.)
d	DNS communication (name resolution), RST/ACK send, UPnP

Table 6: Main transition pattern of each model with proposed method

Traffic pattern	No. of malware samples	representative model	Main transition pattern
0	219	BKDR_IRCBOT	dd
1	79	WORM_DOWNAD	a → a
2	6	WORM_DOWNAD	a → a, Cd → d
3	3	WORM_DOWNAD	b → b
4	2	TROJ_KRYPTIK	c → d, Cd → c
5	2	WORM_DOWNAD	d → d, Ca → a
6	1	WORM_DOWNAD	a → a, Cd → d
7	1	TROJ_MAILBOT	a → a, Cd → d
8	1	WORM_DOWNAD	b → b
9	1	BKDR_SMALL	b → d, Cd → b
10	1	WORM_DOWNAD	d → d, Cc → c
11	1	WORM_ALLPLE	a → a

results are summarized in Table 5. Each code represents one or more traffic features.

The main transition pattern of each model with the proposed method is shown in Table 6.The proposed method created 12 representative models, and each traffic pattern had bias of main transition pattern. In contrast, the one-representative-model method created one representative model, as shown in Table 7. A comparison of the main transition patterns of proposed method with that of the one-representative-model method shows that the latter is included in the main transition patterns of proposed method. It also shows that there is a big difference between the identification rate of the two methods. Therefore, the number of traffic patterns with the latter method is insufficient. A greater number of models is needed to represent the infected traffic. The transition pattern depends on the number of feature classes. If the feature is classified to more complex classes, it is unavoidable that combination explosion will occur. In this study, we set to four codes for vector quantization algorithm.

6.2 Effectiveness of Detection

We discuss the effectiveness of detection by proposed method. As mentioned, the proposed method had the highest identification rate. To evaluate the effectiveness to represent the model of proposed mehotd, we calculated the minimum value of the cumulative minimum distance between each representative model and the target malware traffic sample in the training data. We calculated the mean value of the minimum value of all combinations of representative models and all malware traffic samples in the training data. The shorter the distance between the representative model for each traffic pattern and all malware traffic samples in the training data, the better the representative models represent all malware traffic samples in the training data. We calculated the minimum value of the cumulative minimum distance between each representative model and the malware traffic samples in the testing data. We also calculated the mean value of the minimum value of all combinations of representative models and all malware traffic samples in the testing data. We did the same for the model based on the vendor's definitions and the one representative model. These results are summarized in Table8.

Table 7: Main transition pattern of one representative model

Representative model	Main transition pattern
WORM_DOWNAD	d → d

Table 8: Mean value of minimum distance between representative model and all malware traffic samples

Method	Training data	Testing data
Proposed method (time interval 10 s)	10.34	27.17
Proposed method (time interval 20 s)	3.02	4.45
Proposed method (time interval 30 s)	10.19	13.61
Proposed method (time interval 40 s)	5.30	5.99
Security-vendor-definition-based method (time interval 60 s)	12.67	15.62
One-representative-model method (time interval 60 s)	5603.65	3534.65

The proposed method (20-s time interval) had the minimum distance for the training and testing data. It was about a quarter that of the security-vendor-definition-based method for the training data. It is about 1/1800 that of the one-representative-model method (the method without clustering of malware samples) for the training data. The shorter the cumulative minimum distance, the better the models of infected traffic patterns represent the features of all the traffic. Therefore, the proposed method represents the infected traffic pattern better than the two other methods shown in Table 8.

We investigated the transition pattern of malware samples classified as each traffic pattern. To show the difference of transition pattern, we show a example of the histogram of transition pattern of two worms in Fig. 8 and Fig. 9. In these figures, horizontal axis is transition pattern and vertical axis is ratio of appearance transition pattern. The worm shown in the Fig. 8 is classified as traffic pattern 0. The worm shown in the Fig. 9 is classified as traffic pattern 1.

The outline of the histogram of two worm samples is difference each other. There are many transition patterns of d → d in the Fig. 8. There are many transition patterns of a → a in the Fig. 9. Worm malwares are selected as two traffic patterns. So worm malwares are separated into some groups. The results demonstrate that it need to represent infected traffic data with some traffic patterns.

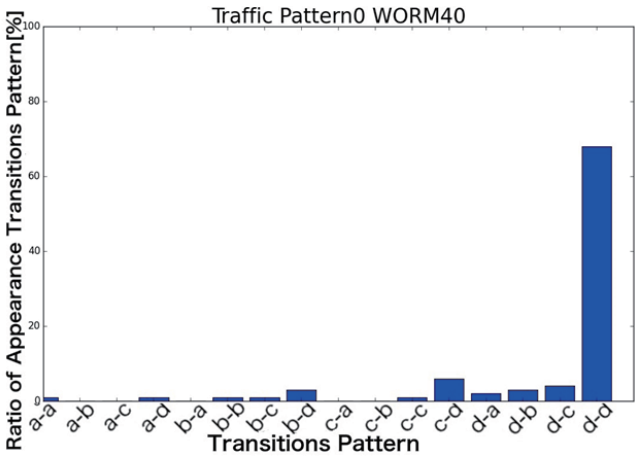


Figure 8: The histogram of worm of traffic pattern 0

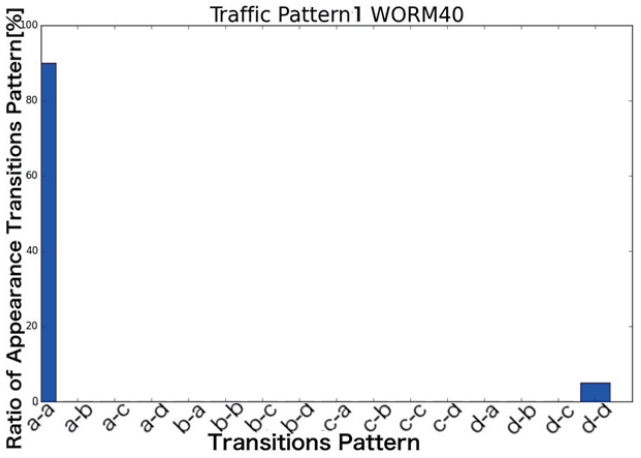


Figure 9: The histogram of worm of traffic pattern 1

6.3 Detection of Unknown Malware

Finally, we discuss the detection of unknown malware. In our experiments, we created models of normal and malware-infected traffic from only training data and used the models to identify malware traffic samples in the testing data. Six classes of ten malwares from the security vendor's definitions were included in the testing data and not in the training data. A malware class corresponds to malware with the same prefix_family name. Malware with the same prefix_family name is considered to be subspecific malware. The ten malware traffic samples represented six malware classes: two were PE_SALITY malware, one was TROJ_KRYPTK malware, one was TROJ_LSADCOM malware, two were TROJ_SPNR malware, three were TROJ_VILSEL malware, and one was TSPY_FAREIT malware. The remaining 190 malware traffic samples were subspecific malware found in the training data. When we focused on the hash value of the malware traffic samples, the training and testing data did not overlap, and the testing data was unknown malware.

As shown In Table 1, the proposed method had an identification rate of 100% for four of the five time intervals. That is, all malware traffic samples in the testing data were correctly identified, including the unknown malware traffic samples of the malware classes included in the training data and the unknown samples of the malware classes not included in the training data. The proposed method is thus able to identify unknown malware samples of a malware class not included in the training data.

7 CONCLUSION

Our method for detecting malware-infected traffic samples is based on the similarity between the pair of malware samples in this paper. Simulation evaluation demonstrated that the proposed method can effectively identify malware-infect traffic with high accuracy.

Future work includes conducting a large-scale experiment to better evaluate the effectiveness of the proposed method.Since normal traffic must be classified as normal when practical, a method for detecting infected traffic combined with a method for detecting normal traffic must be studied. In this paper, we focused on detecting malware infections (including unknown malware infections). Future work includes investigating how to create models of normal traffic for use in classifying unknown normal traffic.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number 15H01684.

REFERENCES

[1] Y. Otsuki, M. Ichino, S. Kimura, M. Hatada, H. Yoshiura, "Evaluating payload features for malware infection detection," Journal of Information Processing (JIP), Vol. 22, No. 2, pp. 376-387 (2014).

[2] K. Kuwabara, H. Kikuchi, M. Terada, M. Fujiwara, "Heuristics for Detecting Types of Infections from Captured Packets," Computer Security Symposium, pp. 1-6 (2009) (in Japanese).

[3] A. Mohaisen, A. G. West, A. Mankin, O. Alrawi, "Chatter: Classifying Malware Families Using System Event Ordering," IEEE Conference on Communications and Network Security, pp. 283-291 (2014).

[4] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, D. Garant, " Botnet detection based on traffic behavior analysis and flow intervals," Computers & Security, Vol. 39, Part A, pp. 2-16 (2013).

[5] M. Piskozub, R. Spolaor, I. Martinovic, "MalAlert: Detecting Malware in Large-Scale Network Traffic Using Statistical Features," ACM SIGMETRICS Performance Evaluation Review, Volume 46, Issue 3, pp. 151-154, (2018).

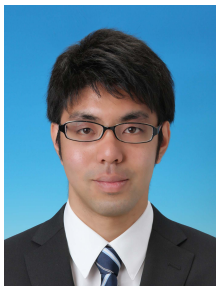
[6] D. Chiba, T. Yagi, M. Akiyama, K. Aoki, T. Hariu, " Design and Evaluation of a Profiling Method to Detect Post-infection Communications," Computer Security Symposium, pp. 960-967 (2014) (in Japanese).

[7] G. Thatte, U. Mitra, J. Heidemann, " Parametric Methods for Anomaly Detection in Aggregate Traffic,"

IEEE/ACM Transactions on Networking, vol. 19, issue 2, pp. 512-525 (2011).
[8] K. Li, R. Chen, L. Gu, C. Liu, J. Yin, "A Method Based on Statistical Characteristics for Detection Malware Requests in Network Traffic," International Conference on Data Science in Cyberspace (DSC), pp. 527-532 (2018).
[9] M. Ichino, K. Kawamoto, T. Iwano, M. Hatada, H. Yoshiura, "Evaluating header information features for malware infection detection," Journal of Information Processing, Vol. 23, No. 5, pp. 603-612 (2015).
[10] T.-F. Yen, M. K. Reiter, "Traffic Aggregation for Malware Detection," 5th International Conference, Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 207-227 (2008).
[11] M. Z. Rafique, J. Caballero, "FIRMA: Malware Clustering and Network Signature Generation with Mixed Network Behaviors," Lecture Notes in Computer Science, Volume 8145 (2013).
[12] T. Ichida, M. Ichino, M. Hatada, N. Komatsu, H. Yoshiura, "A study on malware detection method using feature's state transition," Symposium on Cryptography and Information Security, 1E1-2 (2012) (in Japanese).
[13] G. Nychis, V. Sekar, D. G. Andersen, H. Kim, H. Zhang, "An Empirical Evaluation of Entropy-based Traffic Anomaly Detection," IMC '08 Proceedings of the 8th ACM SIGCOMM conference on Internet measurement, pp. 151-156 (2008).
[14] Y. Linde, A. Buzo, R. Gray, "An Algorithm for Vector Quantizer Design, IEEE Trans. on Commun," Vol. 28, No. 1, pp. 84-95 (1980).
[15] M. Hatada, M. Akiyama, T. Matsuki, T. Kasama, "Empowering Anti-malware Research in Japan by Sharing the MWS Datasets," Journal of Information Processing, pp. 579-588 (2015).
[16] T. Akizuki, M. Ichino, N. Komatsu, K. Takeshita, H. Hasegawa, "A study on External Factors to Estimate Network Traffic," Technical Report of IEICE, Vol. 110, No. 455, CQ2010-70, pp. 19-24 (2011) (in Japanese).

(Received October 19, 2019)
(Revised April 20, 2019)

Masatsugu Ichino received his B.E. degree in electronics, information and communication engineering from Waseda University, Tokyo, Japan in 2003, and M.E. and Ph.D. degrees in computer science and engineering from Waseda University, Tokyo, Japan, in 2005 and 2008, respectively. He is currently an associate professor at the Graduate School of Informatics and Engineering, University of Electro-Communications, Tokyo, Japan. His research interests include biometrics, network security and pattern recognition. He is a member of IEICE and IPSJ.



Yuuki Mori received his M.E. degree in engineering from University of Electro-Communications,Tokyo,Japan in 2019.

Mitsuhiro Hatada is currently a manager at NTT Corporation, and an adjunct researcher at Waseda University. He received his B.E. and M.E. degrees in computer science and engineering, and Ph.D in engineering from the Waseda University in 2001, 2003 and 2018, respectively. He joined NTT Communications Corporation in 2003 and has been engaged in the R&D of applied security on anti-malware and threat intelligence. He is a member of IPSJ and IEICE.

Hiroshi Yoshiura received his B.S. and D.Sc. from the University of Tokyo, Japan in 1981 and 1997. He is currently a Professor in the Graduate School of Informatics, the University of Electro-Communications. Before joining UEC, he had been at Systems Development Laboratory, Hitachi, Ltd. He has been engaged in research on information security and privacy where he has published more than 150 refereed papers and has more than 120 registered patents.