Journal of Information Processing Vol.23 No.5 603–612 (Sep. 2015)

[DOI: 10.2197/ipsjjip.23.603]

Regular Paper

Evaluating Header Information Features for Malware Infection Detection

Masatsugu Ichino[1,a)] Kenji Kawamoto[2] Toru Iwano[1] Mitsuhiro Hatada[3] Hiroshi Yoshiura[1]

Received: November 28, 2014, Accepted: June 5, 2015

Abstract: We analyzed traffic data after a malware infection and clarified which features would be the most effective in the detection of infection. The focus is on the use of traffic data to detect infections and on the use of features that do not change much over time from those of the training data. The characteristics of features that are effective for detecting malware infections are also described. Experimental results clarified the effects of the time difference, and the effective features that were little affected by the time difference were identified. There is thus a need to focus on the effect of the time difference when investigating malware infection detection.

Keywords: malware, infection detection, traffic, header, vector quantization

1. Introduction

The proliferation of Internet use in recent years has increased the threat of malware, a shortened term for malicious software, which can cause harm in our lives by leaking personal information or taking control of our personal computers. The harm caused by malware is increasing in both breadth and depth, and damage from botnets, whose activities are unabated, and infections from Web sites such as Gumblar [1] has increased in recent years. The need for countermeasures is urgent.

This paper concerns infection detection, which we broadly classify as malware detection, intrusion detection and infection detection. Intrusion detection involves 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. The malware infections of recent years have been difficult to notice, and infections have spread widely without users knowing that their computers are being used. Therefore, infection detection for personal computers and upstream machines in the network such as routers and firewalls is also an important measure for preventing the spread of infection.

The research reported here focuses on the use of traffic data to detect infection. This approach determines the features of normal communication traffic and the traffic of infected machines and applies pattern recognition techniques to detect infections. Infection detection based on traffic data uses only the input and output communication traffic of the target machine. Basically, some traffic is generated if there is an infection [2], so this method holds promise as a means of detection from outside the target machine. Furthermore, this method can be applied to the home electronic products and corporate terminals that are currently growing in popularity. Although those devices do not necessarily have sufficient computing power or memory for the introduction and updating of detection modules, malware detection can be performed externally by observing the device's exchange of traffic when it connects to a network. A more promising approach is to externally detect malware infections by observing the device's traffic patterns when it connects to a network, as illustrated in **Fig. 1**.

PCs, home electronic products and corporate terminals and so on are often used for the long term. The traffic data may change year to year because new applications and services of network are occurred and used. It is thus necessary to use features that do not change much over time from those of the training data. However, effective features have not been thoroughly evaluated for malware infection detection. In this paper, we demonstrate the necessary to evaluate the features and clarify the effects of the time difference. And we demonstrate the effectiveness of malware infection detection using fusion of the effective features that were little affected by the time difference were identified.

This paper is organized as follows. Section 2 describes the requirements for a traffic feature to be effective for detecting malware infections. Section 3 describes previous research in this area. Section 4 describes the evaluation of the features, and Section 5 shows the experimental results and demonstrates the effec-

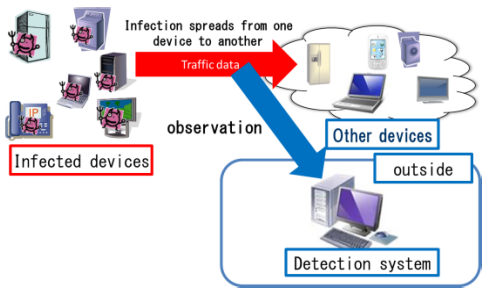


Fig. 1 Malware infection detection from outside PCs, home electronic products and corporate terminals and so on.

[1] The University of Electro-Communications, Chofu, Tokyo 182–8585, Japan
[2] Waseda University, Shinjuku, Tokyo 169–8555, Japan
[3] NTT Communications Corporation, Minato, Tokyo 108–8118, Japan
[a)] ichino@inf.uec.ac.jp

tiveness of fusion of selected feature. Section 6 summarizes the key points and mentions future work.

2. Effective Features for Detecting Malware Infections

The distribution of data is varied by used feature in feature space. Features are effective for detecting malware if they can be used to easily separate normal traffic from malware traffic without any overlap in feature space, as illustrated in **Fig. 2**. Features are ineffective if they cannot be used to easily separate normal traffic from malware traffic by using an algorithm. Therefore, the use of effective features will improve the performance of malware infection detection.

The evaluation metrics were the true positive rate (TPR), i.e., the rate at which malware traffic was correctly classified as malware, and the true negative rate (TNR), i.e., the rate at which normal traffic was correctly classified as normal.

The higher the TPR, the higher the security level because of the reduced chance of misdetecting a malware infection. Generally, a feature with a higher TPR is more effective for detecting a malware infection. However, a high TNR is also important for PCs, home electronic products and corporate terminals because the higher the TNR, the greater the convenience because of the reduced number of false alerts. A false alert causes the user inconvenience because the user must disconnect the device from the network and initiate the malware removal process. A feature with a high TNR is also an effective feature. The overall TPR and TNR can be improved by fusing the features that have a high TPR with features that have a high TNR (including features that have both a high TPR and a high TNR).

If traffic features that change much over time from those of the training data are used for malware infection detection, the TPR and TNR will greatly decrease over time. This increases the risk of malware infection. These decreases in TPR and TNR can be avoided by using traffic features that do not change much over time from those of the training data. This is especially true for PCs, home electronic products and corporate terminals used for a long period of time.

Therefore, features effective for detecting malware infections in PCs, home electronic products and corporate terminals

- have a high TPR and do not change much over time from those of the training data,
- have a high TNR and do not change much over time from those of the training data, and
- have both a high TPR and a high TNR and do not change much over time from those of the training data.

Using a combination of these features improves the effectiveness of detection.

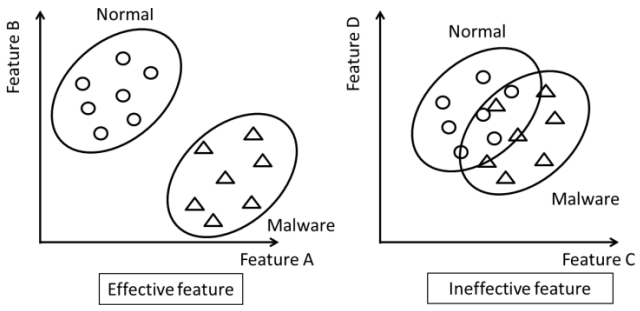


Fig. 2 Results of using effective and ineffective features.

3. Previous Work

Previous studies on malware detection and network intrusion detection included ones that looked at signature-based detection and anomaly-based detection [3].

The most commonly used type of method is signature-based detection. Each file has a unique signature, something like a fingerprint of an executable. A signature-based method uses the patterns extracted from various types of malware. A signature must be prepared for each type of malware. Therefore, a signature-based method cannot detect malware if the signature for the malware is unknown. Anomaly-based detection is also commonly used. An anomaly-based method detects anomalous behavior on a computer or computer network. If the behavior of unknown malware is similar to that of known malware, the unknown malware may be detected. The focus here is on the latter type of method—malware detection based on traffic characteristics.

The following discussion describes the features used in previous research on malware detection and network intrusion detection by methods based on the traffic characteristics.

Hiramatsu studied a clustering method for defining multiple normal states from network traffic data [4]. The normalized numbers of ICMP, SYN, FIN, UDP, TCP except SYN, and FIN packets extracted every 60 minutes are used to define multiple normal states.

Kugisaki focused on the host's transmission intervals as a feature and demonstrated that there is a difference in the transmission interval between human-originated traffic and botnet-originated traffic [5].

Saad focused on detecting bots before they are used to launch an attack. Information about the size of the payloads, the number of packets, the length of duplicate packets, and the number of concurrent active ports used to build a feature set is used by a machine learning algorithm to detect P2P botnets [6].

Garg focused on the traffic characteristics such as the size of the packets, the number of packets, the size of the payloads, and the packet length per flow or host and used a machine learning algorithm to detect P2P botnets [7].

Soniya proposed botnet detection method using header information [8]. Traffic data is cut in 30 minitues and extracted the traffic communication that is not normally used by using destination IP address and port number. For this traffic communication, botnet traffic is detected by using packet inter arrival time and synscan.

Suzuki evaluated the identification rate of changing the length of timeslot [9]. They used three features (mean packet size and so on) for identify the normal traffic and malware traffic.

Previous research did not sufficiently evaluate the features in terms of identification accuracy and the effect of the time difference between the captured data and the training data.

We think that there are two main requirements for effective malware infection detection.

- Detect malware infections with a high TPR and a high TNR.

- Detect malware infection stably over a long term.

To meet the first requirement, we previously focused on malware infection detection using effective features for each type of malware described by Otsuki et al. [10]. Each type of malware exhibits a unique behavior. To give an example, a worm exploits vulnerabilities in software to carry out infections on the network. A Trojan horse, in contrast, accesses a specific site and requests the downloading of unauthorized files, thereby exposing the infected computer to even more threats. It has been shown that a subspecies of a certain malware tends to exhibit the same behaviors. This means that it is possible to identify the features effective for each type of malware. We divided malware into three types—worm, Trojan horse, and file-infected virus—and clarified which features would be most effective for each type.

To meet the second requirement, we propose using a combination of effective features that change little over time to detect malware infection. In this paper, we demonstrate the need to evaluate the features and clarify the effects of the change over time. We also demonstrate the effectiveness of using a combination of effective features that change little over time for malware infection detection.

4. Evaluation Method and Data

Thirty-six features were evaluated in terms of identification accuracy and the effect of the time difference between the captured data and the training data. The effect of the time difference was demonstrated through experimental results.

4.1 Traffic Features

The results of previous research were used as a guide to extracting features from the packet header and compiling statistics about the header information.

Table 1 lists the 36 features evaluated.

Features 1–10 are used for Internet application identification and malware infection detection. They are a basic representation of traffic data.

TCP fragment information is often used for malware detection, and features 11–32 are commonly used for malware detection. Since we did not know whether it was better to use the number or ratio, we evaluated both the number and ratio for the TCP fragment information.

Many malwares cause infection by using the 80/TCP and 443/TCP ports (features 34 and 36) because many users often use these ports. They are thus representative of malware download and command and control (C&C) communication. Therefore, we used features 34 and 36 for evaluation.

The 69/UDP port (feature 33) is used for normal network control. Malwares also use this port (the Blaster worm for example). Therefore, we used feature 33 for evaluation.

Since the mail receiving function is normally set to receive mail on a regular basis, it is possible to misidentify mail receiving as polling communication between the host and C&C server. The 110/TCP port (feature 35) is likely to be reflected in the normal characteristic as representative of normal mechanical operation. Therefore, we used feature 35 for evaluation.

Our goal here is to show the need to use features that are effective for detecting malware infection stably over a long term. In our evaluations, feature 35 is representative of normal operation, and features 33, 34, and 36 are representative of both normal and malware infection operation.

Table 1 Features evaluated.

Number	Feature (unit)
1	number of packets
2	sum of packet sizes (bytes)
3	mean packet size (bytes)
4	minimum packet size (bytes)
5	maximum packet size (bytes)
6	standard deviation of packet size (bytes)
7	mean transmission interval (seconds)
8	minimum transmission interval (seconds)
9	maximum transmission interval (seconds)
10	standard deviation of transmission interval (seconds)
11	number of SYN packets
12	number of FIN packets
13	number of PSH packets
14	number of ACK packets
15	number of RST packets
16	number of URG packets
17	number of SYN/ACK packets
18	number of FIN/ACK packets
19	number of PSH/ACK packets
20	number of RST/ACK packets
21	ratio of SYN packets to TCP packets
22	ratio of FIN packets to TCP packets
23	ratio of PSH packets to TCP packets
24	ratio of ACK packets to TCP packets
25	ratio of RST packets to TCP packets
26	ratio of URG packets to TCP packets
27	ratio of SYN/ACK packets to TCP packets
28	ratio of FIN/ACK packets to TCP packets
29	ratio of PSH/ACK packets to TCP packets
30	ratio of RST/ACK packets to TCP packets
31	number of ICMP packets
32	number of UDP packets
33	number of 69/UDP port packets
34	number of 80/TCP port packets
35	number of 110/TCP port packets
36	number of 443/TCP port packets

4.2 Method

The following describes the method we used for identifying infected traffic and normal traffic.

4.3 Codebook Creation by Vector Quantization

We created an infected codebook and a normal codebook in advance, in which learning was performed using only infected traffic for the former and only normal traffic for the latter. Codebook means the centroid of cluster that is divided from distribution of training data. Since the objective of this study is to evaluate individual features, we created a one-dimensional codebook for each feature. We used the Linde-Buzo-Gray (LBG) + splitting algorithm [11] for vector quantization setting the number of levels to

four types: 2, 4, 8, 16 and 32. We choose four types of parameters because we refer Ref. [11] and number of timeslot we used in this experiment is small.

Vector quantization represents data by using some codebooks, which represent the characteristic of normal or infection traffic data. The LBG splitting algorithm iteratively creates good codebook by applying splitting and representation to the initial codebook.

Description how to calculate the number of codebook is following. We gave the value of frequency of target feature included in each timeslot as following input vector.

Notation: Input vector $x = \{x_0, \cdots, x_{k-1}\}$, k: number of time slot, $d(a,b) = \sum_{i=0}^{k-1} |a_i - b_i|^2$, N_set = number of level, parameter $N = 1$

Algorithm

1. Initialization: Given a distortion threshold ε, and an initial N-level codebook A_0. Set $m = 0$ and $D_{-1} = \infty$
2. Given $A_m = \{y_i;\ i = 1, \cdots, N\}$, find its minimum distortion cluster $P(A_m) = \{S_i;\ i = 1, \cdots, N\} : x \in S_i$ if $d(x, y_i) < d(x, y_j)$ for all j. Compute the resulting average distortion, $D_m = D(\{A_m, P(A_m)\}) = \min_{y \in A_m} d(x, y)$
3. If $(D_{m-1} - D_m)/D_m < \varepsilon$ and $N =$ N_set, halt with A_m and $P(A_m)$ describing final and output the codebook A_m. If $(D_{m-1} - D_m)/D_m < \varepsilon$ and $N <$ N_set, $N = N \times 2$, and Splitting: $y_i = y_i + \delta$, $y_{i+N} = y_i - \delta$ and go to 1. otherwise continue.
4. Find the optimal reproduction codebook $x(P(A_m)) = \{x(S_i);\ i = 1, \cdots, N\}$ for $P(A_m)$. Set $A_{m+1} = x(P(A_m))$. Replace m by $m + 1$ and go to 2.

A_m is codebook calculated by LBG splitting algorithm. We explain this algorithm by using an example (**Fig. 3**).

We calculated the codebook for each cluster by using the timeslot data in each cluster. This operation corresponds to (1) in Fig. 3. Next, we split the codebook by adding and reducing minute values from the target codebook. This operation corresponds to (2). Then, we upgraded the affiliation cluster of the data in each timeslot and calculated the codebooks for each cluster by using the timeslot data in each cluster. This operation corresponds to (3). Operations (2) and (3) were repeated until the convergence condition for the codebooks ($(D_{m-1} - D_m)/D_m < \varepsilon$ and $N =$ N_set) were met.

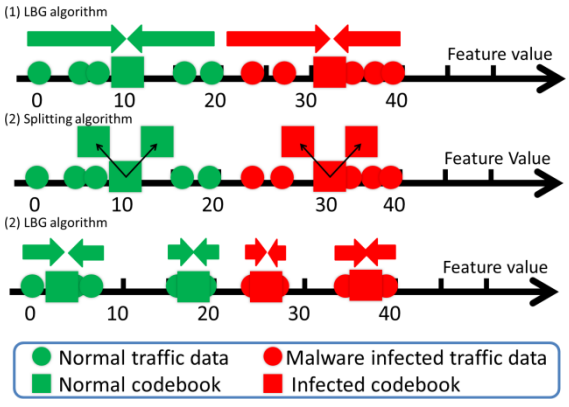


Fig. 3 Vector quantization.

4.4 Computation Method Using Codebooks

To be able to detect malware infection stably over a long term, we need to evaluate the features and clarify the effects of changes in the features over time from those of the training data. There are two requirements for a feature to be effective.

- The distance between the distribution of normal traffic data and the distribution of normal traffic data must be short, and the distance between the distribution of infected traffic data and the distribution of infected traffic data must be short.
- The distance between the distribution of normal traffic data and the distribution of infected traffic data must be long.

We therefore represent the distributions of normal traffic data and of infected traffic data as codebooks by using vector quantization and evaluate the efficiency of the features by calculating the distance between the codebook and target traffic data. It is appropriate to represent the traffic data distribution by using codebooks because there are a number of conditions related to traffic data. As a basic study, we used the nearest distance classifier to calculate the distance between the codebook and target traffic data.

Using vector quantization and the above codebooks, we calculated the distance between test data and the infected/normal codebooks and identified the test data as being infected or normal depending on which distance from the infected/normal codebooks was shorter.

4.5 Evaluation of Feature

We therefore studied features with both a high true positive rate (TPR) and a high true negative rate (TNR), features with only a high TPR, and features with only a high TNR. TPR is the rate at which infected traffic is correctly classified into infected category. TNR is the rate at which normal traffic is correctly classified into normal category. Description to calculate TPR and TNR is below.

$$\text{TNR} = \frac{\text{Number of the infected test data classified correctly into infected category}}{\text{Number of the total infected test data}}$$

$$\text{TPR} = \frac{\text{Number of the normal test data classified correctly into normal category}}{\text{Number of the total normal test data}}$$

An overview of the evaluation is shown in **Fig. 4**.

The TNR and TPR were calculated for each feature and parameter for each timeslot by using traffic data from 2009, 2010, and 2011.

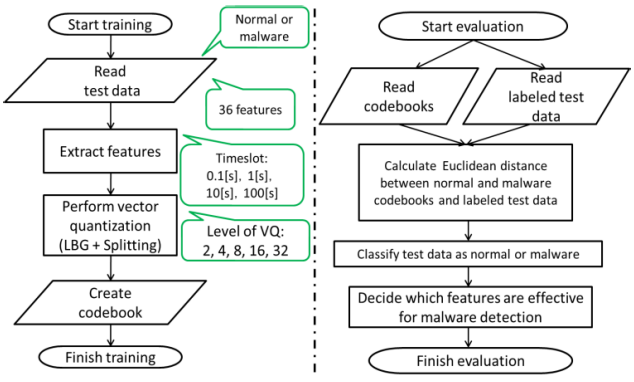


Fig. 4 Overview of evaluation.

4.6 Data

The malware traffic data came from the Cyber Clean Center (CCC) DATAset [12] by capturing it in a honeypot system, and the normal traffic was taken from the Intranet. Both types of data were captured on the same dates: March 13, 14, and 15, 2009. The CCC2009 dataset was used for the malware codebook. The normal traffic data was used for the normal codebook. The test data for the malware traffic came from the CCC2009, CCC2010, and CCC2011 datasets, and the test data for the normal traffic came from 2009, 2010, and 2011.

Ideally, the normal and malware traffic data would have been captured under the same circumstance. However, resources on malware traffic are limited. In addition, normal traffic data captured in a honeypot system would not be realistic because nobody generates traffic in a honeypot. This problem was overcome by preprocessing the normal traffic data so that it imitated the capture circumstances of the malware traffic.

• Preprocessing for normal traffic

The normal traffic data was preprocessed to meet the following requirements.

1. Generated from one host—It is necessary to imitate the capture circumstances of malware traffic.
2. Generated by normal users—If the host is infected with malware, it will download or update new malware or try to connect to the Internet. However, such transmissions are normal in terms of their behavior. In this research area, it is important to be able to distinguish malware transmissions from those of people with no malicious intent. Hence, the traffic generated by a normal user must be used.

• Preprocessing for malware traffic

The honeypot traffic data used (from CCC2009, CCC2010, and CCC2011) included scan traffic, exploit traffic, and infected traffic. This means it also included non-infected traffic data. However, it was essential to use only infected traffic data in the evaluation. Hence, preprocessing to extract the malware traffic from the other attack traffic data was done. The procedure for doing so is as follows.

1. Remove control packets generated only in a honeypot situation.
2. Divide the pcap data in the OS reset interval of the honeypot.
3. Check whether traffic is truly infected by referring to the malware collection log provided in the CCC DATAset and look for the first packet of the malware transmission.
4. Extract the traffic data after the first packet of the malware transmission.

5. Evaluation Results

5.1 Evaluation in Terms of Capture Time

This section summarizes the experimental results and analyzes the features that were effective in detecting malware on the basis of the time difference between the captured data and the training data. The features are classified into those for which the effect of the time difference was significant and those for which the effect was insignificant. Then, which features overall were the most effective are summarized. Finally the effectiveness of fusion of the most effective features is evaluated.

Table 2 Average TPR and TNR.

Year captured for test data	2009	2010	2011
Average TPR	57.0%	54.1%	51.2%
Average TNR	36.1%	35.2%	40.7%

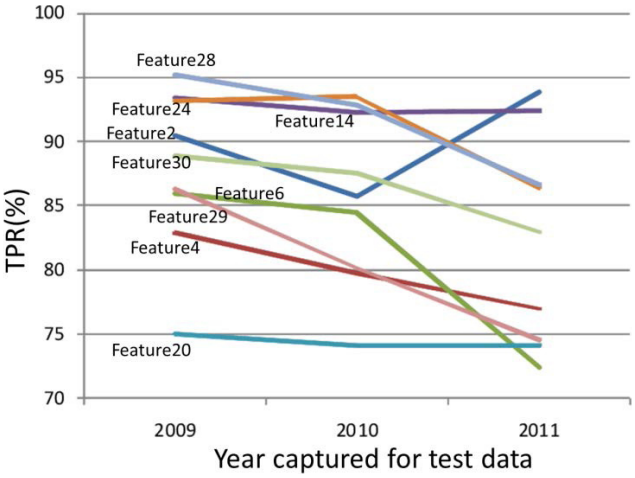


Fig. 5 TPRs of features for which average TPR was higher than 70% for all three years.

Table 3 TPRs over 90% over three years for minimum packet size.

Timeslot length (s)	vector quantization level	year captured for test data		
		2009	2010	2011
0.1	32	99.8%	94.6%	90.2%
1	32	100%	99.6%	95.8%
10	32	94.0%	92.0%	92.4%

First, the changes in the TPR and TNR over the course of three years are evaluated.

Table 2 shows the average TPRs and TNRs in 2009, 2010, and 2011. The average TPR or TNR is the average of the corresponding values calculated for each feature, timeslot length, and number of vector quantization levels.

The average TPR in 2011 was the lowest, while the average TNR in 2011 was the highest. This means that the time difference affects the TPR and TNR.

5.2 Analysis in Terms of Capture Time

5.2.1 Analysis on Basis of TPR

A "low TPR" feature, i.e., a feature for which the effect of the time difference is small, is not appropriate for detecting malware infections. So "high TPR" features were selected.

The TPRs of the features for which the average TPR was higher than 70% for all three years are plotted in **Fig. 5**.

From these features, those for which the effect of the time difference was small were selected: features 4, 14, 20, and 30 had the smallest changes in TPR. Three of these features had average TPRs higher than 80%: feature 4 (minimum packet size), feature 14 (number of ACK packets), feature 20 (number of RST/ACK packets). These three features were analyzed from the viewpoint of separating normal traffic from malware traffic.

1. Minimum packet size (feature 4)

As shown in **Table 3**, the TPRs over 90% over the three years for minimum packet size were for timeslot lengths of 0.1, 1, and

Table 4 Average and standard deviation of minimum packet size in normal and malware test traffic data (unit for "average" is bytes per slot).

Timeslot length (s)	normal/ malware	year captured	average	standard deviation
0.1	normal	2009	79.5	95.6
		2010	81	113.1
		2011	93.7	134.7
	malware	2009	68.6	32.4
		2010	84.7	89.1
		2011	114.4	174.6
1	normal	2009	60.1	1.3
		2010	60.8	7
		2011	60	0
	malware	2009	70.7	31.1
		2010	73.3	34.8
		2011	101.1	83.1
10	normal	2009	60.9	0.2
		2010	60.1	3
		2011	60	0
	malware	2009	67.8	28.2
		2010	71.2	40.4
		2011	102.2	113.9
100	normal	2009	60.4	2.8
		2010	60	0
		2011	60	0
	malware	2009	69.3	44.9
		2010	66.8	40.4
		2011	84.2	60.4

10 seconds for quantization level 32.

As shown in **Table 4**, the minimum packet size for normal traffic was almost always 60 bytes when the timeslot interval was larger than 1 second. In contrast, the minimum packet size for malware traffic varied considerably. There was an enormous difference in the standard deviation between the minimum packet size of normal traffic and that of malware traffic. For normal traffic, the standard deviation was almost always 0 while it was much larger than zero for malware traffic. This difference means that minimum packet size is effective for malware detection.

2. Number of ACK packets (feature 14)

As shown in **Table 5**, the number of ACK packets for the malware test traffic was much smaller that for the normal test traffic. This means that the number of ACK packets is an effective feature for detecting malware infections.

3. Ratio of RST/ACK packets to TCP packets (feature 20)

In terms of the ratio of RST/ACK packets to TCP packets, the TPR was high and the effect of the time difference was small. However, both the malware test traffic and the normal test traffic had many timeslots in which the ratio of RST/ACK packets to TCP packets was 0. Moreover, the TPR was high because the malware codebook was closer to 0 than the normal codebook, and timeslots in which the ratio of RST/ACK packets to TCP packets is 0 are classified as malware. Therefore, the ratio of RST/ACK packets to TCP packets cannot be used to accurately detect malware. This feature is thus not useful for malware detection.

Table 5 Average number of ACK packets in normal and malware test traffic.

Timeslot length (s)	normal/ malware	year captured	average
0.1	normal	2009	10.9
		2010	6.9
		2011	3.3
	malware	2009	2.6
		2010	0.6
		2011	2.3
1	normal	2009	174.1
		2010	20.0
		2011	19.0
	malware	2009	3.1
		2010	1.3
		2011	3.4
10	normal	2009	289.1
		2010	103.2
		2011	787.2
	malware	2009	10.0
		2010	10.9
		2011	11.8
100	normal	2009	1305.4
		2010	432.8
		2011	6669.8
	malware	2009	40.7
		2010	73.5
		2011	37.5

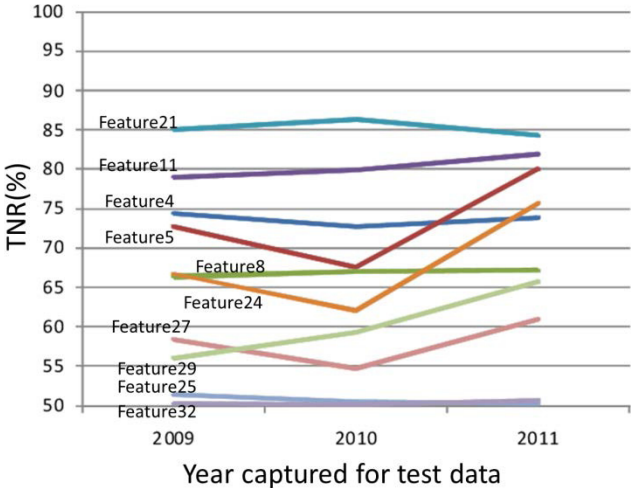


Fig. 6 TNRs of features for which average TNR was higher than 50% for all three years.

5.2.2 Analysis in Terms of TNR

A "low TNR" feature, i.e., a feature for which the effect of the time difference is small, is not appropriate for detecting malware infections. So "high TNR" features were selected.

The TNRs of the features for which the average TNR was higher than 50% for all three years are plotted in **Fig. 6**.

From these features, those for which the effect of the time dif-

Table 6 TNRs over 90% over three years for minimum packet size.

timeslot length (s)	vector quantization level	year captured for test data		
		2009	2010	2011
0.1	4	98.4%	92.6%	90.4%
	8	98.2%	92.6%	93.3%
1	4	99.8%	98.7%	100.0%
	8	100.0%	98.7%	100.0%
	16	98.0%	99.1%	100.0%
10	2	100.0%	100.0%	100.0%
	4	100.0%	100.0%	100.0%
	8	100.0%	100.0%	100.0%
	16	100.0%	100.0%	100.0%
100	2	99.3%	100.0%	100.0%
	4	100.0%	100.0%	100.0%
	8	100.0%	100.0%	100.0%
	16	100.0%	100.0%	100.0%

ference was small were selected: features 4, 8, 11, 21, 25, and 32 were the least affected by the time difference.

Three of these features had average TNRs higher than 70%: feature 4 (minimum packet size), feature 11 (number of SYN packets), and feature 21 (ratio of SYN packets to TCP packets). These three features were analyzed from the viewpoint of separating normal traffic from malware traffic.

1. Minimum packet size (feature 4)

As shown in **Table 6**, the TNRs over 90% over the three years for minimum packet size were for various timeslot lengths and quantization levels.

The average and standard deviations of the minimum packet sizes are shown in Table 4.

As mentioned in the TPR analysis, the minimum packet size differs between normal traffic and malware traffic, so the minimum packet size is effective for detecting malware.

2. Number of SYN packets; ratio of SYN packets to TCP packets (features 11 and 21)

TNR was very high when the number of SYN packets or the ratio of SYN packets to TCP packets was used. This is because malware traffic data tends to resemble the data in a SYN scan. Therefore, the values in a malware codebook are much larger than those in the normal codebook. Moreover, normal test traffic data does not have many SYN packets. Therefore, almost all of the normal traffic data were classified as normal. That is why the TNR was very high. However, malware traffic does not always have SYN scan data. Although these features are not particularly useful for classifying traffic as normal or malware, they would be effective for predicting or detecting an attack.

5.3 Correlation among Features

We discuss the correlation among the features.

We calculated the correlation coefficients for all combinations (${}_{36}C_2$) of two features by using

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

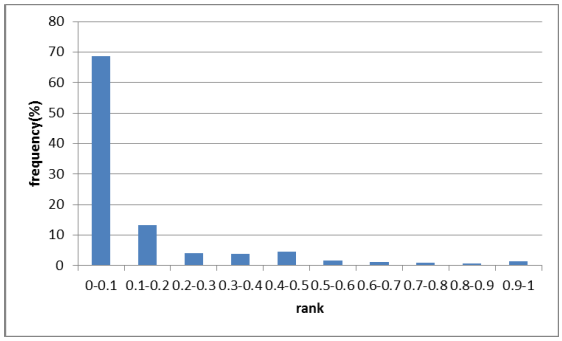


Fig. 7 Histogram of correlation coefficients.

Table 7 Feature combinations with correlation coefficient ≥ 0.90.

combination of feature		correlation coefficient
maximum packet size (bytes)	standard deviation of packet size (bytes)	0.99
mean packet size (bytes)	standard deviation of packet size (bytes)	0.98
mean packet size (bytes)	maximum packet size (bytes)	0.96
sum of packet sizes (bytes)	number of ACK packets	0.94
number of UDP packets	number of 69/UDP port packets	0.90
mean transmission interval (seconds)	minimum transmission interval (seconds)	0.90

where r is the correlation coefficient, x_i, y_i are the features, n is the number of timeslots, and $\bar{x}$, $\bar{y}$ are the mean values of the two features.

We calculated the correlation coefficients using traffic data combining normal and malware traffic data, as described in Section 4.6. A histogram of the coefficients is shown in **Fig. 7**.

The horizontal axis represents the rank of the correlation coefficients. The vertical axis represents the frequency rate of the number of correlation coefficients corresponding to each rank. Many of correlation coefficients were small, so there was little correlation between many features. The six combinations with fairly strong coefficients (≥ 0.90) are listed in **Table 7**.

The difference between two features that are strongly correlated is the amount of statistical data. The basic type of two features is the same (e.g., packet size). The larger the packet size, the greater the standard deviation in packet size roughly and the greater the mean packet size roughly. The standard deviation is related to the maximum and mean packet sizes. That is, the maximum packet size, mean packet size, and standard deviation are strongly correlated. Moreover, the greater the number of ACK packets, the larger the sum of the packet sizes roughly. There is thus a strong correlation between the sum of the packet sizes and the number of ACK packets. None of the features was strongly correlated with all the other features.

We applied principal component analysis (PCA) to the traffic data and used feature vectors (36 dimensions) consisting of 36 features normalized by each feature's variance. First, we investigated the cumulative contribution ratio of the principal components. The results are plotted in **Fig. 8**.

The horizontal axis represents the number of principal components, and the vertical axis represents the cumulative contribution ratio (%). The lack of a principal component with a very high cumulative contribution ratio in this figure demonstrates that there were no dominant principal components.

We also mapped the feature vectors (36 dimensions) in a two-dimensional space by using PCA and investigated the traffic data distribution. The traffic data distribution is shown in **Fig. 9**.

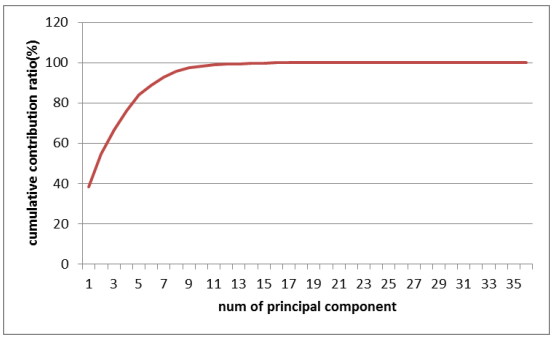


Fig. 8 Cumulative contribution ratio vs. number of principal components.

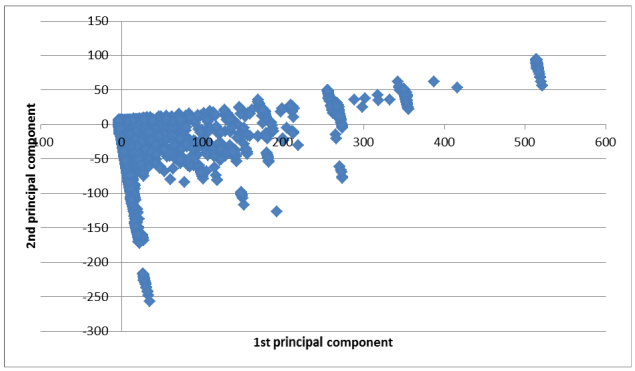


Fig. 9 Two-dimensional feature space.

The horizontal axis represents the 1st principal component, and the vertical axis represents the 2nd principal component. The spread of the distribution in various directions indicates that there was no dominant component.

These results are convincing evidence that dominant features do not exist.

5.4 Evaluation of Effective Features for Malware Infection Detection

As mentioned in Section 2, features that are effective for malware detection are those that are little affected by the time difference between the captured data and the training data and those that have both a high TPR and a high TNR. Features that are little affected by the time difference and that have either a high TPR or a high TNR may also be effective. The minimum packet size, the number of SYN packets, the ratio of SYN packets to TCP packets, and the number of ACK packets were identified as the most effective features for malware detection on the basis of the requirements described in Section 2.

Ideally, the normal and malware traffic data would have been captured under the same circumstance. However, resources on malware traffic are limited. In addition, normal traffic data captured in a honeypot system would not be realistic because nobody generates traffic in a honeypot. This problem was overcome by preprocessing the normal traffic data so that it imitated the capture circumstances of the malware traffic.

To make our evaluation as reasonable as possible, normal traffic data were preprocessed to remove packets corresponding to the routing within the LAN, thereby removing packets with environment-specific characteristics. Likewise, the malware traffic data were preprocessed to remove packets corresponding to the specific data capture environment of the CCC DATASET honeypots.

Table 8 Identification rate using fusion of features.

case	algorithm	year captured for test data		difference in identification rate between 2010 and 2011
		2010	2011	
1	decision tree	99.8%	89.8%	10.0%
	naïve Bayes	73.4%	60.4%	13.0%
2	decision tree	99.9%	91.1%	8.8%
	naïve Bayes	89.0%	86.4%	2.6%

Since we needed to use only malware-infected traffic data (without any normal traffic data) as the malware-infected traffic data for our evaluation, we extracted the malware-infected traffic data from the traffic data in each CCC DATASET on the basis of the correspondence between the traffic data and logs.

The four features we identified as being effective for detecting malware infection may not be effective in other traffic environments because traffic data (including noisy data) depend on the environment in which they are captured. Our goal here is to simply show the need to use features effective for detecting malware infection stably over a long term. For detecting malware infection in other environments, normal traffic data for the target environment should be captured in advance, and malware-infected traffic data should be obtained from a distribution organization such as MWS [12]. And it is necessary to evaluate features effective for detecting malware infection stably over a long term as explained in Section 2.

5.5 Malware Infection Detection Using Several Features

In addition to evaluating the effectiveness of individual features, the effectiveness of the fusing features was evaluated.

The identification rates in two cases were compared. The "identification rate" is defined as the ratio of correct identification of normal traffic data and of malware traffic data.

In case 1, malware infection was detected by using a fusion of all the features listed in Table 1. In case 2, malware infection was detected by using a fusion of the minimum packet size, the number of SYN packets, the ratio of SYN packets to TCP packets, and the number of ACK packets.

The same dataset shown in Section 4.6 was used. Vectors that concatenated the features were used for input. CCC2010 was used for malware traffic training data. Normal traffic data captured in 2010 were used for normal traffic training data. CCC2011 was used for malware traffic test data. Normal traffic data captured in 2011 were used for input data test data. Decision tree and naïve Bayes classifiers, which are often used as traffic identification algorithms [13], [14], were used as traffic identification algorithms.

As shown in **Table 8**, the differences in the identification rates between 2010 and 2011 in case 2 were smaller than those in case 1. This means that it needs to detect malware infection using features that are little affected by the time difference between the captured data and the training data.

Previous work did not sufficiently evaluate the features in terms

of the effect of the change over time between the features of the testing data and those of the training data. We clarified the effects of the change over time and demonstrated that malware infection detection was improved by using a combination of the effective features that changed less over time.

Case 1 corresponds to previous work (effective features not identified and used) while case 2 corresponds to our approach (identify and use effective features). As shown in Table 8, the identification rate for 2010 with naïve Bayes for case 2 was better than that for case 1. In contrast, the identification rates with decision tree were virtually the same. The identification rate for 2011 with naïve Bayes for case 2 was again better than that for case 1. The identification rate with decision tree for case 2 was slightly better than that for case 1. The difference in identification rate between 2010 and 2011 with decision tree for case 2 was better than that for case 1. The difference in identification rate between 2010 and 2011 with naïve Bayes for case 2 was better than that for case 1. Our approach thus produces better results than previous approaches.

6. Conclusion

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. Requirements for effective features for detecting malware infections were identified. Experimental results demonstrated the effects of the time difference. Then the effective features that were little affected by the time difference were described. This study revealed the need to focus on the effect of the time difference when investigating malware infection detection.

There are infection activities specific to each type of malware, such as “Internet connection confirmation” for worms, “download malware order to attack communications” for Trojan horses, and “IRC connection” for file-infected viruses. Future work includes investigating malware infection detection by focusing on the traits of each malware type, meeting the two main requirements (detect malware infection stably over a long term and detect malware infections with a high TPR and a high TNR), and evaluating the features of ports other than those investigated in this study as those features may also be effective.

References

[1] Mendyk-Krajewska, T. and Mazur, Z.: Problem of Network Security Threats, *IEEE Conference on Human System Interactions*, pp.436–443 (May 2010).

[2] Hatada, M., Inazumi, T., Arikawa, J. and Tanaka, Y.: A Case Study on Light-weight URL Blacklist Generation based on Sandbox Analysis, *IEICE Technical Report*, Vol.114, No.117, pp.309–314 (2014) (in Japanese).

[3] Idika, N. and Mathur, A.P.: A Survey of Malware Detection Techniques, Technical Report, Department of Computer Science, Purdue University (2007).

[4] Hiramatsu, N., Waizumi, Y., Tsunoda, H. and Nemoto, Y.: Using Multiple Normal States for Network Anomaly Detection, *IEICE Technical Report*, CS2006-32, pp.61–66 (2006) (in Japanese).

[5] Kugisaki, Y., Kasahara, Y., Hori, Y. and Sakurai, K.: Study for botnet detection based on behavior observation of data transmission interval, *Symposium on Cryptography and Information Security* (2009) (in Japanese).

[6] Saad, S., Traore, I., Ghorbani, A., Sayed, B., Zhao, D., Lu, W., Felix, J. and Hakimian, P.: Detecting P2P botnets through network behavior analysis and machine learning, *Annual International Conference on Privacy, Security and Trust*, pp.174–180 (2011).

[7] Garg, S., Singh, A.K., Sarje, A.K. and Peddoju, S.K.: Behaviour Analysis of Machine Learning Algorithms for detecting P2P Botnets, *IEEE International Conference on Advanced Computing Technologies*, pp.1–4 (2013).

[8] Soniya, B. and Wilscy, M.: User Traffic Profile for Traffic Reduction and Effective Bot C&C Detection, *International Journal of Network Security*, Vol.16, No.1, pp.46–52 (2014).

[9] Suzuki, N., Koike, A., Suzuki, T. and Miyaho, N.: Study of the parameters extraction method for achieving the cyber attack detection, *The 76th National Convention of IPSJ*, pp.3-625–3-626 (2014).

[10] Otsuki, Y., Ichino, M., Kimura, S., Hatada, M. and Yoshiura, H.: Evaluating payload features for malware infection detection, *Journal of Information Processing*, Vol.22, No.2, pp.376–387 (2014).

[11] Linde, Y., Buzo, A. and Gray, R.M.: An Algorithm for Vector Quantization, *IEEE Trans. Commun.*, Vol.28, No.1, pp.84–95 (1980).

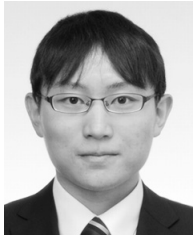
[12] Hatada, M., Nakatsuru, I. and Akiyama, M.: Datasets for Anti-Malware Research – MWS 2011 Datasets, *Anti Malware Engineering WorkShop 2011*, pp.1–5 (2011) (in Japanese).

[13] Livadas, C., Walsh, R., Lapsley, D. and Strayer, W.T.: Using Machine Learning Techniques to identify botnet traffic, *IEEE LCN Workshop on Network Security*, pp.967–974 (2006).

[14] Stevanovic, M. and Pedersen, J.M.: An efficient flow-based botnet detection using supervised machine learning, *IEEE International Conference on Computing, Networking and Communications*, pp.797–801 (2014).

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 assistant professor at the Graduate School of Informatics and Engineering, the University of Electro-Communications, Tokyo, Japan. His research interests include biometrics, network security, quality of service, and pattern recognition. He is a member of IEICE, IPSJ and IEEE.

Kenji Kawamoto received his B.E. degree in electronics, computer science and engineering from Waseda University, Tokyo, Japan in 2011, and M.E. in computer science and engineering from Waseda University, Tokyo, Japan, in 2013. His research interests include network security.



Toru Iwano received his B.E. degree in engineering from University of Electro-Communications, Tokyo, Japan in 2014. He is a master’s degree student at the Graduate School of Informatics and Engineering, University of Electro-Communications.

Mitsuhiro Hatada was born in 1978. He is currently a Ph.D. student with particular interest in anti-malware. He received his B.E. and M.E. degrees in computer science and engineering from Waseda University in 2001 and 2003, respectively. He joined NTT Communications Corporation in 2003 and has been engaged in the R&D of network security and anti-malware. He is a member of IEICE and IPSJ.

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 and received the President's Technology Award from Hitachi in 2000, the Best Paper Award from Information Processing Society of Japan in 2005 and 2011, the Industrial Technology Award from the Institute of Systems, Control and Information Engineers in 2005, the Best Paper Award of IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing in 2006, and the Best Paper Award from Japan Society of Security Management. He is a member of IEICE, IPSJ, JSSM, ISCIE and IEEE.