Electronic Preprint for Journal of Information Processing Vol.22 No.2

Recommended Paper

Evaluating payload features for malware infection detection

Yusuke Otsuki[1,a)] Masatsugu Ichino[1] Soichi Kimura[1] Mitsuhiro Hatada[2]
Hiroshi Yoshiura[1]

Received: June 6, 2013, Accepted: November 1, 2013

Abstract: Analysis of malware-infected traffic data revealed the payload features that are the most effective for detecting infection. The traffic data was attack traffic using the D3M2012 dataset and CCC DATAsets 2009, 2010, and 2011. Traffic flowing on an intranet at two different sites was used as normal traffic data. Since the type of malware (worm, Internet connection confirmation, etc.) affects the type of traffic generated, the malware was divided into three types—worm, Trojan horse, and file-infected virus—and the most effective features were identified for each type.

Keywords: malware, infection detection, traffic, payload

1. Introduction

The Internet has become an essential part of work and everyday life. It continues to expand and become increasingly convenient, but it is also a source of problems because the harm done by malware to the benefits of the Internet also continues to expand. "Malware" is short for "malicious software," that is, software designed with ill intent in mind. It has become a threat to our everyday life by infecting personal computers, resulting in data corruption, leaked personal information, and various other problems.

The number of malware incidents increased by about 20 times over the four-year period from 2007 to 2011 and exceeded 2.5 million for the first time in survey history in 2011 [1].

In response to this trend, security vendors have been developing anti-malware software for detecting and removing malware using a signature-based detection technique [2]. This detection technique, however, must prepare a signature for each item of malware reflecting its features, meaning that it is unable to detect unknown malware appearing in great quantities over a short period of time. There is therefore a need for a detection technique based on the likelihood of infection that can detect an unknown malware infection in the early stage of its spread.

This infection detection requires that the target traffic data be classified as either normal or infected, which is a two-class task. The malware infection detection thus needs to classify the target network traffic into normal or infected so that the detection system can use the results to judge whether the target host is infected.

Our target data is mainly traffic data coming from an infected host. Our approach is to use payload information because there is much information available in addition to header data that can be used to discriminate between infected traffic and normal traffic.

In this study, we first analyzed existing research on infection detection based on payload information. We found that the previous studies did not identify payload features that would make it easy to discriminate between a normal and an infected system. Once the effectiveness of individual features is clarified, methods using an optimal combination of these features can be developed.

With this objective in mind, we analyzed malware-infected traffic data and identified the payload features that would be the most effective in detecting infection. The traffic data was attack traffic using the D3M2012 dataset and CCC DATAsets 2009, 2010, and 2011 [3] (referred to below as CCC2009, CCC2010, and CCC2011). Traffic flowing on an intranet at two different sites was used as normal traffic data. Since the type of malware (worm, Internet connection confirmation, etc.) affects the type of traffic generated, we divided the malware into three types consisting of Trojan horse, worm, and file infected virus and identified the features that would be most effective for each type.

The rest of this paper is organized as follows. In Section 2, we summarize the payload features commonly used in related work and describe the goal of our research. In Section 3, we describe our experimental method along with the specific features to be evaluated in this paper. Section 4 then presents the results of our experiment evaluating the ease of discriminating between infected traffic and normal traffic for each feature, and Section 5 considers which features have the highest discrimination rate for each type of malware. Section 6 presents the results from evaluate several important combinations of features including word occurrence such as "connection" and "Cookie." Section 7 concludes

[1] The University of Electro-Communications, Chofu, Tokyo 182–8585, Japan
[2] NTT Communications Corporation, Minato, Tokyo 108–8118, Japan
[a)] otsuki@uec.ac.jp

The initial version of this paper was presented at CSS2012 held between October 30 and November 1 in 2012, which was sponsored by SIGCSEC. This paper was recommended to be submitted to Journal of Information Processing (JIP) by the chairman of SIGCSEC.

the paper.

2. Related Work

2.1 Features Used in Previous Research

Previous research related to malware infection detection and infected network detection used various payload features.

The appearance frequency rate has been used in several studies [4], [5], [6]. Kuwabara et al. determined from payload information in bot-generated traffic that there are characteristic strings (such as exe and NICK) corresponding to different types of malware behavior, and they used those strings as features [6].

Lu et al. [7] proposed a technique that focuses on traffic data carrying bot-generated remote-control communications. This technique examines the packet payload information of normal traffic and of bot-generated infected (abnormal) traffic and treats the appearance frequency rate (number of bytes) of ASCII character codes within the payload as a feature.

Other infection detection techniques [8], [9] handle the problem of unknown attacks by using a decision tree, with the HTTP request length and total HTTP request size used as features.

In short, previous studies have used the appearance frequency rate of ASCII character codes, the appearance frequency rate of character strings, and the HTTP request length as features for detecting malware infection. None of these studies, however, evaluated the effectiveness of individual features.

2.2 Research Goal

The goal of this research topic from a long-term view is to improve malware detection accuracy.

Because so much new malware appears every day, it is difficult to totally prevent infection. Malware infections have become more difficult to detect, so infections have spread widely without users knowing that their computers have been infected. Therefore, malware detection is an important measure for preventing its spread. The accuracy of detection is below the level needed to reduce the spread of malware, so improving accuracy is important.

There are two essential subjects that need to be completed to achieve this goal. Although they are not independent, we can still focus on either of them.

(1) Find traffic features that differ between traffic from infected computers and traffic from uninfected computers. The greater the difference, the more effective the feature.

(2) Develop algorithms that use these features to classify traffic as either infected or not infected.

Previous research focused on developing algorithms using features based only on the researchers' experience. The most unique point of our research is its focusing on finding effective traffic features.

The main point of our approach was exhaustiveness; i.e., we evaluated 261 features, including all occurrences of all ASCII code. Because we used so many features, we mainly evaluated each feature independently rather than evaluating combinations of features, which would require evaluating 261^n combinations for an n-gram. However, we did evaluate several important combinations of features including word occurrence, i.e., sequences of ASCII code features.

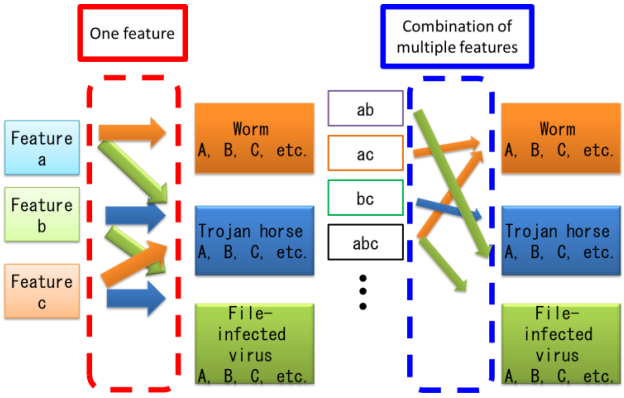


Fig. 1 Research approach.

As shown by the Trend Micro Security database [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 accesses a specific site and requests the downloading of unauthorized files, thereby exposing the infected computer to even more threats. A subspecies of similar malware tends to exhibit the same behavior [10]. This means that it is possible to identify effective features for each type of malware. We thus divided malware into three types—worm, Trojan horse, and file-infected virus—and identified the features that would be the most effective for each type.

Figure 1 illustrates our approach. Here, the term "One Feature" means that we find features that differ between traffic streams that are infected and those that are not infected for each type of malware independently. Here, the term "Combination of multiple features" means that we evaluate combinations of features for each type of malware.

3. Methodology

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

3.1 Codebook Creation by Vector Quantization

We created an infected codebook and a normal codebook in advance. They were used respectively for infected traffic training and normal traffic training. A codebook is the centroid of a cluster that is divided from the distribution of the training data. Since the objective of this study was to evaluate individual features, we created a one-dimensional codebook for each feature. We used the Linde-Buzo-Gray (LBG) + splitting algorithm [12] for vector quantization, with the number of levels set to 2, 4, 8, or 16, because we refer [12] and number of timeslot we used in this experiment is small.

Vector quantization yields a data description in terms of clusters or groups of data points that possess strong internal similarities. Vector quantization uses the sum of the square distances from the cluster centers (codebooks in this paper) and seeks the grouping that minimizes the sum of the square distances. Each resulting group forms a cluster, with data points in the same cluster being more similar to each other than to ones in other clusters.

The range of each codebook was calculated as follows. The number of appearances of a target feature in each timeslot was

input as a vector:
Notation: Input vector $x = \{x_0, \cdots x_i, \cdots, x_{k-1}\}$, where k is the number of time slots, x_i is i the time slot.
square distortion $d(a,b) = \sum_{i=0}^{k-1} |a_i - b_i|^2$,
N_set = number of levels, and 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 the final quantizer and output codebook A_m. If $(D_{m-1} - D_m)/D_m < \varepsilon$ and $N <$ N_set, $N = N \times 2$, and Codebook 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 with $m + 1$ and go to 2.

A_m is codebook calculated using the LBG splitting algorithm.

3.2 Computation Method Using Codebooks

There are three requirements for a feature to be effective.
(1) The distance between data distribution in normal traffic should be short.
(2) The distance between data distribution in infected traffic should be short.
(3) The distance between data distribution in normal traffic and data distribution in infected traffic should be long.

The first two mean that the variance for each traffic data type should be small. The last one means that normal traffic and infected traffic should have little overlap.

We thus need to investigate the relationship between normal and infected traffic to evaluate the effectiveness of individual features. We did this by representing the distributions of normal traffic and of infected traffic as codebooks obtained using vector quantization (LBG + splitting algorithm). Then, by calculating the distance between the codebooks and the target traffic data, we measured the goodness of separation between normal and infected traffic. We used the nearest distance classifier for the classification of the target network traffic to calculate the distance between each codebook and the target traffic data. The support vector machine (SVM), AdaBoost, deep learning, and extreme learning machine (ELM) methods are not appropriate for evaluation based on the three requirements because they basically do not measure the goodness of separation. Instead, they determine the boundary between two category distributions in feature space.

Using vector quantization and these codebooks, we calculated the distance between the test data and the infected and normal codebooks and identified the test data as being infected or normal in accordance with the distance.

3.3 Feature Evaluation

We identified the 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 as infected. TNR is the rate at which normal traffic is correctly classified as normal.

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

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

Since the goal of this research is to improve the accuracy of detecting malware infection of a host, we need to discriminate between normal traffic data and infected traffic data. To evaluate the effectiveness of discrimination, we used TPR and TNR.

Each timeslot of the tested traffic data containing the target feature (for example ASCII code "i") was discriminated between normal and infected. The number of timeslots with correctly classified data was then calculated. The TPR and TNR were then calculated using the equations above.

To evaluate each feature's effectiveness, we focused on the position relation between the distribution of normal traffic and that of infected traffic. We calculated the distance between the two distributions and made a histogram to identify overlap between the distributions (shown in Section 5). A discussion from the viewpoints of the behavior for each type of malware is included in Appendix B.

4. Experiment

4.1 Time Slots

We used time slots to extract features from traffic data. Time slots divide traffic data into intervals lasting a specific length of time.

Dividing time-varying traffic into time slots and representing that traffic in units of time slots enables one to identify normal and infected traffic by focusing on the overall temporal variation of that traffic. We set the time-slot duration to 1 s and identified the features for every time slot.

4.2 Payload Features

We used the entire payload included in traffic of communications over HTTP protocol using 80/tcp port because the dataset included in payload is only communications over HTTP protocol using 80/tcp port in this experiment. We evaluated the 261 features frequently used in previous research:

- appearance frequency of 255 ASCII codes
- appearance frequency of five characteristic character strings (GET, POST, exe, whatismyip, and checkip)
- HTTP request length.

4.3 Experimental Data

To evaluate the differences in data-acquisition environments, we used traffic flowing on an intranet at two different sites (intranet A and intranet B) as normal traffic. This enabled us to evaluate features without them being easily affected by differences in the data-acquisition environment. We divided the data from each site into training data for creating a normal codebook and test

Table 1 Malware type(s) and malware sample(s) for each data set.

Malware types	CCCDATA set2011	CCCDATA set2010	CCCDATA set2009	D3M2012
Worm	WORM_DOWNAD.AD	WORM_DOWNAD.AD WORM_MAINBOT.AH WORM_MAINBOT.FY WORM_PALEVO.SMD WORM_PALEVO.BL	WORM_SWTYMLAI.CD	
Trojan horse			TROJ_BUZUS.AGB	TROJ_GEN.R47C7C2 Trojan.Generic_KD.578000 Trojan.Generic_KD.410743 Trojan-Dropper.Win32.Dapato.aoxn
File-infected virus		PE_VIRUT.AV PE_VIRUT.XV	PE_BOBAX.AK	

data.

We used the D3M2012, CCC2009, CCC2010, and CCC2011 datasets as infected traffic data and divided each into training data for creating an infected codebook and test data. The malware samples for each type of malware were divided up fairly evenly among these data sets. For example, three of the four datasets contained six worm samples. Three were used as training data, and three were used as test data. In this way, we were able to determine whether training using three types of worm could enable detection of three other types of worm.

Note that the CCC2009, CCC2010, and CCC2011 datasets included data for communications prior to malware infection. Thus, given the purpose of our evaluation, we extracted from the infected traffic only the traffic following malware infection using the method described by Kawamoto et al. [11]:

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

We used worms, Trojan horses, and file-infected viruses as the three types of malware. The file-infected viruses infected executable files with the .exe extension. The names of the malware samples in the CCC DATAset were the names entered in the log files of malware traffic data, and those in the D3M2012 dataset were the names designated by G Data Software, Trend Micro, and Kaspersky Lab using the hash values of those samples. The malware in each of the datasets is listed in **Table 1**.

Table 2 shows the 1-s time slots of normal traffic for training and testing. **Table 3** shows the 1-s time slots of infected traffic for training and testing. We used the data in the first 4,000 time slots on one day as normal traffic training data and the data in the first 4,000 time slots of another day as normal traffic testing data. The training and testing of infected traffic was divided so that the number of malware sample for training and the number of malware sample for testing were the same.

Table 2 Number of 1-s time slots in normal traffic.

	training (slots)	testing (slots)
intranet A	4000	4000
intranet B	4000	4000

Table 3 Number of 1-s time slots in infected traffic.

	training (slots)	testing (slots)
Worm	403	174
Trojan horse	101	105
File-infected virus	287	297

4.4 Experimental Results

Since we use payload features to identify whether the target traffic is infected or normal, we can improve the identification rate by using more effective features. It can also be improved by combining features. We therefore identified features with both a high TPR and a high TNR, features with only a high TPR, and features with only a high TNR.

4.4.1 Features with High TPR and TNR

From the viewpoint of evaluating the features less susceptible to the acquisition environment and to evaluate the effect of the number of quantization levels, we evaluated the average TPR and TNR by using four different quantization levels (2, 4, 8, 16) for each of the 261 features using the normal traffic data (intranets A and B).

Table 4 shows the top 15 features in terms of average TPR and

Table 4 Top 15 features in terms of average TPR and TNR by malware type by intranet.

Malware types	Feature	TPR average of intranet A	TPR average of intranet B	Feature	TNR average of intranet A	TNR average of intranet B
Worm	HTTP request length	100%	99%	ASCII code ">"	89%	79%
	ASCII code "i"	99%	98%	ASCII code "!"	88%	78%
	ASCII code "f"	98%	97%	ASCII code "ETB"	88%	70%
	ASCII code "C"	98%	92%	ASCII code "J"	85%	83%
	ASCII code "E"	97%	94%	ASCII code "H"	84%	74%
	ASCII code "e"	97%	91%	ASCII code "#"	84%	78%
	ASCII code "a"	96%	91%	ASCII code "."	84%	76%
	ASCII code "0"	95%	97%	ASCII code "B"	84%	74%
	ASCII code "f"	95%	91%	ASCII code "E"	84%	75%
	ASCII code "CR"	95%	91%	ASCII code "P"	84%	75%
	ASCII code "/"	95%	92%	ASCII code "R"	84%	74%
	ASCII code ":"	95%	94%	ASCII code "S"	84%	73%
	ASCII code "c"	95%	93%	ASCII code "i"	83%	83%
	ASCII code "s"	95%	94%	ASCII code "M"	83%	75%
	ASCII code "m"	95%	91%	ASCII code "N"	83%	77%
Trojan horse	HTTP request length	100%	100%	ASCII code "<"	85%	72%
	ASCII code "NL*"	100%	100%	ASCII code "US"	85%	76%
	ASCII code "CR"	100%	100%	ASCII code ">"	85%	41%
	ASCII code "0"	100%	100%	ASCII code "¥"	83%	73%
	ASCII code "5"	100%	100%	ASCII code "VT"	82%	56%
	ASCII code "C"	100%	100%	ASCII code "]"	82%	68%
	ASCII code "d"	100%	100%	ASCII code "HT"	81%	65%
	ASCII code "e"	100%	100%	ASCII code "SOH"	80%	56%
	ASCII code "t"	100%	100%	ASCII code "~"	80%	69%
	ASCII code "x"	100%	100%	ASCII code "'"	80%	67%
	ASCII code "A"	100%	100%	ASCII code "FS"	80%	63%
	ASCII code "o"	100%	100%	ASCII code "ESC"	79%	62%
	ASCII code "r"	100%	100%	ASCII code "ACK"	79%	63%
	ASCII code "1"	100%	100%	ASCII code "$"	79%	65%
	ASCII code "2"	100%	100%	ASCII code "EOT"	79%	60%
File-infected virus	HTTP request length	100%	100%	ASCII code "DC2"	92%	68%
	ASCII code "p"	99%	99%	ASCII code "DC3"	90%	68%
	ASCII code "B"	99%	75%	ASCII code "ETB"	89%	62%
	ASCII code "S"	98%	98%	ASCII code "#"	89%	67%
	ASCII code "e"	98%	98%	ASCII code "S"	89%	69%
	ASCII code "T"	98%	94%	ASCII code "Y"	88%	76%
	ASCII code "I"	98%	98%	ASCII code "'"	87%	79%
	ASCII code "o"	97%	98%	ASCII code "M"	87%	79%
	ASCII code "H"	96%	94%	ASCII code "R"	86%	79%
	ASCII code "X"	95%	74%	ASCII code "US"	86%	69%
	ASCII code "W"	95%	93%	ASCII code ")"	85%	75%
	ASCII code "r"	95%	94%	HTTP request length	82%	84%
	ASCII code "G"	95%	78%	ASCII code "("	82%	78%
	ASCII code "N"	95%	80%	ASCII code "f"	82%	73%
	ASCII code "q"	95%	94%	ASCII code "j"	82%	75%

Table 5 TPR and TNR for ASCII code "i" for worm by intranet.

	TPR				TNR			
Quantization level	2	4	8	16	2	4	8	16
Intranet A	100%	100%	100%	100%	82%	80%	81%	81%
Intranet B	100%	100%	100%	94%	88%	86%	88%	86%

Table 6 TPR and TNR for "HTTP request length" for file-infected virus by intranet.

	TPR				TNR			
Quantization level	2	4	8	16	2	4	8	16
Intranet A	98%	99%	98%	98%	83%	80%	86%	83%
Intranet B	97%	98%	98%	97%	83%	80%	80%	82%

TNR when changing the quantization levels by type of malware by intranet. The TPR and TNR values were highly stable for ASCII code "i" for worm and "HTTP request length" for file-infected virus regardless of the acquisition environment. TPR and TNR changed vector quantization levels using the ASCII code of "i" and "HTTP request length, as shown in **Tables 5** and **6**. For both intranets, these features were detected stably with a TPR of more than 94% and a TNR of more than 80% even when the number of quantization levels was changed.

4.4.2 Features with Only High TPR

In Table 4, the features with only a high TPR (enclosed by a thick dotted line) effective for detecting a worm were "HTTP request length" and ASCII codes of "0" and "f"; those effective for detecting a Trojan horse were "HTTP request length" and ASCII codes of "NL*," "CR," "0," "5," "A," "C," "M," "d," "e," "r," "t," and "x"; those effective for detecting a file-infected virus were ASCII codes of "S," "e," "I," "o," and "p." For both intranets, these features were detected stably with a TPR of more than 95% even when the number of quantization levels was changed.

4.4.3 Features with Only High TNR

In Table 4, the feature with only a high TNR (enclosed by a thin dotted line) effective for detecting a worm was ASCII code "J." For both intranets, this feature was detected stably with a TNR of more than 80% even when the number of quantization levels was changed.

5. Discussion

We clarified the effectiveness of the features identified as described in Section 4 by checking the overlap between the normal traffic and infected traffic distributions. In this section, we focus on worm malware and discuss the situation of a histogram generated by appearance frequency. Similar discussions for Trojan horse and file-infected virus malware are found in Appendix A.

5.1 Appearance Frequency of ASCII Code

We made histograms of the appearance frequency of each identified feature. In these histograms, the horizontal axis is the appearance frequency (number of times) and the vertical axis is the ratio of the number of slots containing the target feature to the number of total slots of infected traffic (or normal traffic).

For example, the dark bars in **Fig. 2** show the appearance frequency rate of ASCII code "i" to the total number of slots in the infected traffic data. The leftmost one (appearance frequency 0. . . 9) shows that the slots containing 0–9 ASCII code "i" per slot represent 75% of the total slots in the infected traffic data. The light bars show the appearance frequency rate of ASCII code "i" to the total number of slots in the normal traffic data. **Figure 3** is a continuation of Fig. 2.

Figures 2 and 3, which show examples of features that have been accurately identified, are histograms of the appearance frequency of ASCII code "i" in worm-infected traffic and in normal traffic (intranet A). They show the results of setting the quantization level to 2. The dark bars in Figs. 2 and 3 are to the left of 59 on the x axis, meaning that the appearance frequency of ASCII code "i" in the infected traffic data was less than 59 per slot. The light bars to the right of 1,000 on the x axis indicate that the appearance frequency of ASCII code "i" in the normal traffic data was more than 1,000 per slot. We therefore created a small-value infected codebook and a large-value normal codebook.

One codebook is made for a place where the frequency of ASCII code "i" is high (the number for the infected codebook is 9). The other is made for a place where the codebook number is large (the number for the normal codebook is 1,450). Each codebook represents an outline of a histogram, so using the LBG splitting algorithm was appropriate for calculating the codebook numbers in this experiment.

The dark bars in Figs. 2 and 3 are to the left of 59 on the x axis, meaning that the appearance frequency of ASCII code "i" in the infected traffic data was less than 59 per slot. The light bars to the right of 1,000 on the x axis indicate that the appearance frequency of ASCII code "i" in the normal traffic data was more than 1,000 per slot. We therefore created a small-value infected codebook and a large-value normal codebook.

One codebook is made for a place where the frequency of ASCII code "i" is high (the number for the infected codebook is 9). The other is made for a place where the codebook number is large (the number for the normal codebook is 1,450). Each codebook represents an outline of a histogram, so using the LBG splitting algorithm was appropriate for calculating the codebook numbers in this experiment.

There are two explanations for the normal traffic having

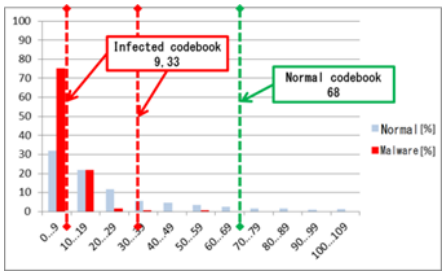


Fig. 2 Histogram for ASCII code "i" (ratio of number of displayed slots to total number of slots is more than 1%). (Horizontal: appearance frequency (no. of times); vertical: appearance frequency rate (%)).

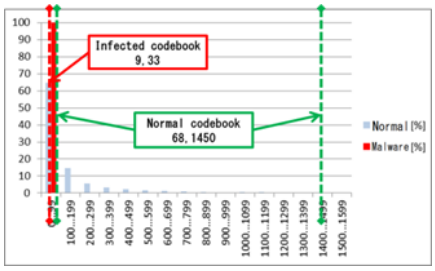


Fig. 3 Continuation of histogram for ASCII code "i" (ratio of number of displayed slots to total number of slots is less than 1%). (Horizontal: appearance frequency (no. of times); vertical: appearance frequency rate (%))

a greater appearance frequency of ASCII code "i." First, the size of the payload in normal traffic is greater than that of the payload in infected traffic. In fact, the byte count of the payload in normal traffic was twice that of the payload in infected traffic. This is why the appearance frequency of ASCII code "i" is so high.

The second explanation is that the appearance of a keyword containing ASCII code "i" in normal traffic is more frequent than that of a keyword containing ASCII code "i" in infected traffic. These keywords include terms containing an ASCII code "i" such as "Cookie," which saves personal information such as login information and so on in normal traffic, and "Content-type images," which are used for multimedia (images and video) appearing in normal traffic data. Character strings containing ASCII code "i," their meaning, and our comments are summarized in **Table 7**.

The findings for Trojan horse and file-infected virus are similar (see Appendix A.).

5.2 Appearance Frequency of HTTP Request Length

A histogram of HTTP request length in normal traffic (intranet A) and in worm-infected traffic infected when the quantization level was set to 2 is shown in **Figs. 4** and **5**. The horizontal axis is the HTTP request length, and the vertical axis is the ratio of the number of slots contained in the range of HTTP request lengths to the total number of slots. Figure 4 shows the range up to a request length of 200, and Fig. 5 shows the range up to 5,000.

The appearance frequencies for the infected traffic were less than 110 while most of those for the normal traffic data were more than 110.

Normal traffic data (user communications) features many types of communication, so the HTTP request length varies. The HTTP request length for much of the infected traffic data was small, and the value of the infected codebook was also small, so the TPR

Table 7 Character strings containing ASCII code "i," their meaning, and our comments.

String	Meaning	Examination	Frequency (Normal/Malware)
Accept-Encoding gzip	The encoding system that the browser is supporting. The server sends the compressed data encoding that supports (such as "gzip").	In order to collect the information, such as confirming the Internet connection, the information amount of packets is small, and malware does not communicate a large amount of information to perform gzip compression.	58%/0% 48%/0%
Connection Keep-alive	To conduct a persistent connection, and to send and receive multiple files in one session.	Malware communicates necessary minimum information with unnecessary issue of multiple request/response.	62%/22% 52%/0%
Cookie	Leave personal information on a PC.	Malware does not communicate with leaving personal information such as login.	31%/0%
Content-type image application	Show the file type of the data of the payload.	Malware does not need to connect frequently to the site that contains images in order to perform simple communication such as confirming the Internet connection.	22%/0% 13%/0%

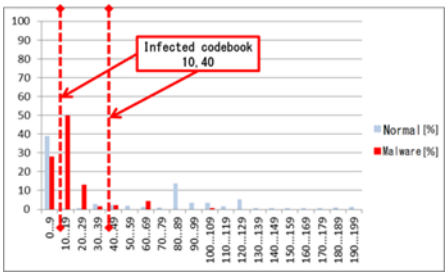


Fig. 4 Histogram for HTTP request length (ratio of number of displayed slots to number of total slots is more than 1%). (Horizontal: request length, vertical: rate of appearance frequency (%)).

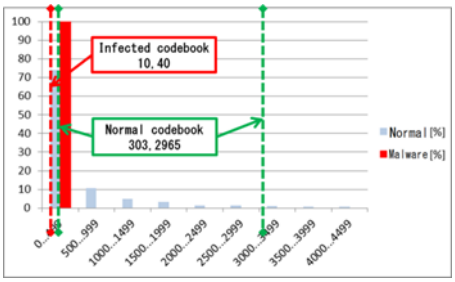


Fig. 5 Continuation of histogram for HTTP request length (ratio of displayed slots to the total number of slots is less than 1%). (Horizontal: request length, vertical: rate of appearance frequency (%)).

was high. We concluded that the TNR was lower because some HTTP requests in normal traffic are shorter.

The results for Trojan horse and file-infected virus were similar (see Appendix A).

6. Evaluation of Words

We evaluated the effectiveness of the words mentioned in the previous section under the same experimental conditions. Those with a high TPR (enclosed by a thick dotted line in **Table 8**) effective for detecting a worm were "Accept-Encoding," "keep-alive," "Cookie," and "Content-type"; those effective for Trojan horse were "Accept-Encoding," "gzip," "keep-alive" and "Cookie"; those effective for file-infected virus were "Accept-Encoding," "gzip," "Connection," "keep-alive," "Cookie," and "Content-type."

7. Conclusion

We evaluated the payload features of three malware types—worm, Trojan horse, and file-infected virus—to determine their effectiveness in detecting malware infection. We used the D3M2012, CCC2009, 2010, and 2011 datasets for infected traffic data and the traffic data of two intranets as normal traffic data.

We found that the TPR and TNR for each malware type using a specific ASCII code and HTTP request length were high and that specific ASCII codes and HTTP request lengths are effective features for detecting infection. We varied the quantization level and focused on the top 15 features in terms of average TPR and TNR and found that when TPR was more than 95% and TNR was more than 80%, the detection rate was "high."

We also identified 3 optimal features for worms, 15 for Trojan horses, and 5 for the file-infected viruses as features that have only a high TPR. Only one feature for worms had a high TNR. As a feature with both a high TPR and high TNR, ASCII code "i" was especially effective for detecting file-infected viruses, and HTTP request length was especially effective for detecting worms.

In addition, 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 the file-infected viruses. We identified the relationships between the identified features and these infection activities.

Now that we have identified the most effective features for

Table 8 Average TPR and TNR for words in each intranet.

Malware types	Feature	TPR average of intranet A	TPR average of intranet B	Feature	TNR average of intranet A	TNR average of intranet B
Worm	Accept-Encoding	100%	100%	Accept-Encoding	20%	50%
	gzip	85%	85%	gzip	60%	59%
	Connection	36%	25%	Connection	56%	55%
	keep-alive	100%	100%	keep-alive	86%	37%
	Cookie	100%	100%	Cookie	85%	70%
	Content-Type	100%	100%	Content-Type	76%	50%
	image	52%	75%	image	51%	62%
	application	76%	75%	application	93%	63%
Trojan horse	Accept-Encoding	100%	100%	Accept-Encoding	20%	50%
	gzip	100%	100%	gzip	60%	59%
	Connection	38%	25%	Connection	45%	42%
	keep-alive	100%	100%	keep-alive	86%	37%
	Cookie	100%	100%	Cookie	85%	70%
	Content-Type	50%	63%	Content-Type	3%	22%
	image	85%	96%	image	89%	59%
	application	75%	75%	application	91%	49%
File-infected virus	Accept-Encoding	100%	100%	Accept-Encoding	20%	53%
	gzip	100%	100%	gzip	60%	58%
	Connection	96%	96%	Connection	28%	51%
	keep-alive	100%	100%	keep-alive	86%	44%
	Cookie	100%	100%	Cookie	85%	65%
	Content-Type	100%	98%	Content-Type	76%	53%
	image	35%	76%	image	29%	62%
	application	79%	75%	application	94%	65%

detecting malware infection for three types of malware, we will next investigate effective combinations of features (n-gram and so on) for discriminating between normal and infected traffic. We will also improve our classification algorithm so that normal traffic and infected traffic can be discriminated more effectively.

References

[1] Gdata Malware Report, Half-yearly Report, January – June 2011, available from ⟨http://www.gdatasoftware.com/uploads/media/G_Data_MalwareReport_H1_2011_EN.pdf⟩.

[2] Fujiwara, M., Terada, M., Abe, T. and Kikuchi, H.: Study for the classification of malware by infection activities, *IPSJ CSEC*, No.21, pp.177–182 (Mar. 2008) (in Japanese).

[3] Hatada, M., Nakatsuru, I. and Akiyama, M.: Datasets for Anti-Malware Research — MWS 2012 Datasets, *MWS2012* (Oct. 2012) (in Japanese).

[4] Yonahara, A., Ohtani, H., Baba, T. and Inada, T.: A Consideration of Spyware Detection using Traffic Analysis, *IPSJ CSEC*, No.70, pp.23–29 (July 2005) (in Japanese).

[5] Kloft, M. et al.: Automatic feature selection for anomaly detection, *Conference on Computer and Communications Security* (2008).

[6] Kuwabara, K., Kikuchi, H., Terada, M. and Fujiwara, M.: Heuristics for Detecting Types of Infections from Captured Packets, *MWS2009* (Oct. 2009) (in Japanese).

[7] Lu, W. et al.: Automatic Discovery of Botnet Communities on Large-Scale Communication Networks, *The 4th International Symposium on Information, Computer, and Communications Security* (2009).

[8] Hareesh, I. et al.: Anomaly detection system based on analysis of packet header and payload histograms, *ICRTIT 2011* (2011).

[9] Yamada, A., Miyake, Y., Takemori, K. and Tanaka, T.: Machine Learning Based IDS with Automatic Training Data Generation, *Trans. IPS Japan*, No.46, pp.1947–1958 (July 2005) (in Japanese).

[10] Trend Micro Security Database, in Japanese, available from ⟨http://jp.trendmicro.com/jp/home/index.html⟩.

[11] Kawamoto, K., Ichida, T., Ichino, M., Hatada, M. and Komatsu, N.: A study of feature evaluation considering effects of year for malware detection, *MWS2011* (Oct. 2011) (in Japanese).

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

Appendix

A.1 Discussion of Trojan Horse and File-infected Virus

A.1.1 Trojan Horse

A.1.1.1 Appearance Frequency of ACII Characters

As an example of the features that were judged accurately in Section 4, we show histograms of the appearance frequency of the ASCII code “o” of the infected traffic data of a Trojan horse and of the normal traffic data of intranet A (**Figs. A·1** and **A·2**). The vertical and horizontal axes of these figures are the same as those in Figs. 2 and 3.

The appearance frequency of ASCII code “o” for the infected traffic data was less than 20 per slot, as shown in Figs. A·1 and A·2. However, there are light bars to the right of 1,000 on the x axis, which indicates that a small-value infected codebook and a large-value normal codebook were created.

There are two reasons that normal traffic had a greater appearance frequency for ASCII code “o.” First, the amount of payload contained in normal traffic is greater than that in infected traffic. In fact, the byte count of the payload in the normal traffic was twice that of the payload in the infected traffic. This explains the frequent appearance of ASCII code “o.”

Second, keywords containing ASCII code “o” appear more often in normal traffic than in infected traffic. These keywords include “Cookie,” which saves personal login information in normal traffic, and “Connection,” which sends and receives multiple files in one session. Character strings containing ASCII code “o,” their meaning, and our comments are summarized in Table 6.

Table A·1 Character strings containing ASCII "o," their meaning, and our comments.

String	Meaning	Examination	Frequency (Normal/Malware)
Accept-Encoding	The encoding system that the browser is supporting. The server sends the compressed data encoding that supports.	In order to collect the information, such as confirming the Internet connection, the information amount of packets is small, and malware does not communicate a large amount of information to perform gzip compression.	58%/0%
Connection	To conduct a persistent connection, and to send and receive multiple files in one session.	Malware communicates necessary minimum information with unnecessary issue of multiple request/response.	62%/50%
Cookie	Leave personal information on a PC.	Malware does not communicate with leaving personal information such as login.	31%/0%
Content-type image application	Show the file type of the data of the payload.	Malware does not need to connect frequently to the site that contains images in order to perform simple communication such as confirming the Internet connection.	13%/0%

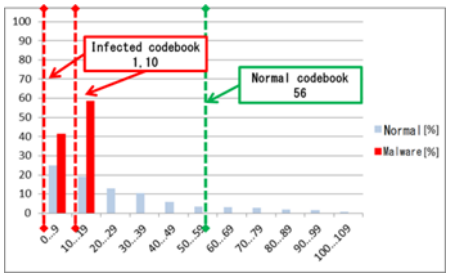


Fig. A·1 Histogram for ASCII code "o" (ratio of number of displayed slots to total number of slots is more than 1%). (Horizontal: appearance frequency (no. of times), vertical axis: rate of appearance frequency (%)).

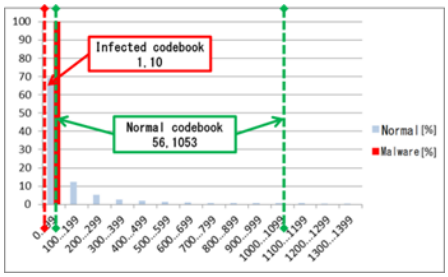


Fig. A·2 Continuation of histogram for ASCII code "o" (ratio of number of displayed slots to total number of slots is less than 1%). (Horizontal: appearance frequency (no. of times), vertical axis: rate of appearance frequency (%)).

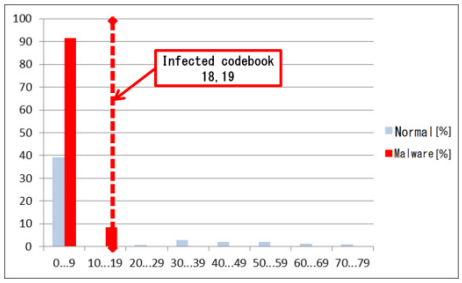


Fig. A·3 Histogram for HTTP request length (ratio of number of displayed slots to total number of slots is more than 1%). (Horizontal: request length, vertical: rate of appearance frequency (%)).

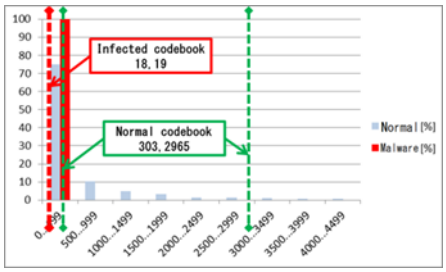


Fig. A·4 Continuation of histogram for HTTP request length (ratio of displayed slots to the total number of slots is less than 1%). (Horizontal: request length, vertical: rate of appearance frequency (%)).

A.1.1.2 Appearance Frequency of HTTP Request Length

Histograms of the appearance frequency for HTTP request length for normal traffic (intranet A) and the infected traffic of a Trojan horse are shown in **Figs. A·3** and **A·4**. The vertical and horizontal axes are the same as those in Figs. 4 and 5.

These figures show the results when the quantization level was set to 2. The appearance frequency for the infected traffic was less than 20, and that for the normal traffic data was mostly more than 100.

Normal traffic data (user communications) features many types of communication, so the HTTP request length varies. The HTTP request length of significantly infected traffic data is small and is classified into the infected codebook. Therefore, TPR was higher and TNR was lower because some HTTP requests were shorter in the normal traffic.

Table A·2 Character strings containing ASCII "i," their meaning, and our comments.

String	Meaning	Examination	Frequency (Normal/Malware)
Accept-Encoding gzip	The encoding system that the browser is supporting. The server sends the compressed data encoding that supports.	In order to collect the information, such as confirming the Internet connection, the information amount of packets is small, and malware does not communicate a large amount of information to perform gzip compression.	58%/0%
Connection	To conduct a persistent connection, and to send and receive multiple files in one session.	Malware communicates necessary minimum information with unnecessary issue of multiple request/response.	62%/50%
Cookie	Leave personal information on a PC.	Malware does not communicate with leaving personal information such as login.	31%/0%
Content-type image image	Show the file type of the data of the payload.	Malware does not need to connect frequently to the site that contains images in order to perform simple communication such as confirming the Internet connection.	22%/0%

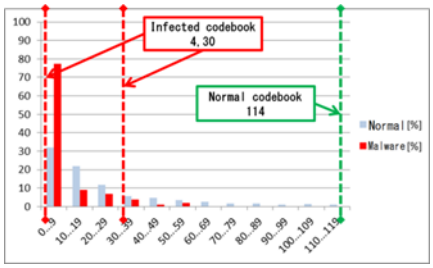


Fig. A·5 Histogram for ASCII code "i" (ratio of number of displayed slots to total number of slots is more than 1%). (Horizontal: appearance frequency (no. of times), vertical: rate of appearance frequency (%)).

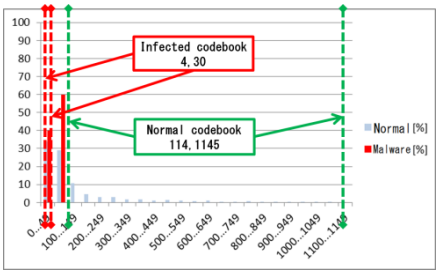


Fig. A·6 Histogram for ASCII code "i" (ratio of displayed slots to the total number of slots is less than 1%). (Horizontal: appearance frequency (no. of times), vertical: rate of appearance frequency (%)).

A.1.2 File-infected Virus

A.1.2.1 Appearance Frequency of ACII Characters

As an example of the features that were judged accurately in Section 4, we show histograms of the appearance frequency of ASCII code "i" of the infected traffic data of a file-infected virus and of normal traffic data (intranet A) in **Figs. A·5** and **A·6**). The vertical and horizontal axes of these figures are the same as those in Figs. 2 and 3.

The appearance frequency of ASCII code "i" of the infected traffic data was less than 59 per slot, as seen in Figs. A·5 and A·6.

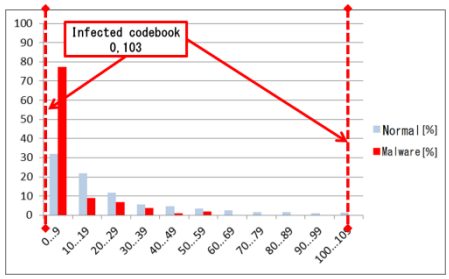


Fig. A·7 Histogram for HTTP request length (ratio of number of displayed slots to total number of slots is more than 1%). (Horizontal: request length, vertical: rate of appearance frequency (%)).

There are light bars to the right of 1,000 on the x axis, which indicates that a small-value infected codebook and a large-value normal codebook were created.

There are two reasons that normal traffic had a greater appearance frequency for ASCII code "i" First, the size of the payload in the normal traffic was greater than that in the infected traffic. In fact, the byte count of the payload in the normal traffic was twice that of the payload in the infected traffic. This is why ASCII code "i" appeared so frequently.

The second reason is that keywords containing ASCII code "i" occur more frequently in normal traffic than in infected traffic. These keywords include "Cookie," which saves personal login information in normal traffic, and "Content-type image," which is used for multimedia (images and video) appearing in normal traffic. Character strings containing ASCII code "i," their meaning, and our comments are summarized in **Table A·2**.

A.1.2.2 Appearance Frequency of HTTP Request Length

Histograms of the appearance frequency of HTTP request length in normal traffic (intranet A) and the infected traffic of a file-infected virus are shown in **Figs. A·7** and **A·8**. The vertical and horizontal axes of these figures are the same as those in

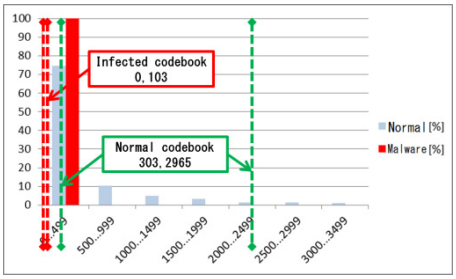


Fig. A·8 Continuation of histogram for HTTP request length (ratio of displayed slots to the total number of slots is less than 1%). (Horizontal: request length, vertical: rate of appearance frequency (%)).

Figs. 4 and 5.

These figures show the results when the quantization level was set to 2. The appearance frequency for the infected traffic was less than 20, and that the appearance frequency for the normal traffic was mostly more than 110.

Normal traffic data (user communications) features many types of communication, so the HTTP request length varies. The HTTP request length of significantly infected traffic data is small and is classified into the infected codebook. Therefore, TPR was higher and TNR was lower because some HTTP requests were shorter in the normal traffic.

A.2 Discussion of Malware Behavior by Type

As mentioned, we investigated the behavior of each type of malware and identified the features most effective against each type of malware. Here, we describe the payload information for normal and infected traffic data for each type (worm, Trojan horse, and file-infected virus), explain the associated behaviors, and discuss the most effective features for detecting infection.

A.2.1 Worm

• Determine Internet connection

With worm infections, the worm first checks that the target PC is connected to the Internet and then performs activities to infect the PC. It then checks the Internet connection to a particular domain, i.e., a site that displays the user's IP address, such as www.whatismyipaddress.com and checkip.dyndns.org.

• Download malware in order to attack communications

After the worm completes the infection, which serves as a starting point, it downloads other malware from individual servers by using "HTTP GET" commands such as "*GET /vss.exe HTTP/1.0*" and "*GET /fdc1.data HTTP/1.0.*" The HTTP request length of infected traffic data is generally shorter than 110, while that of normal traffic data is generally longer than 110 (i.e., the ratio of the number of slots featuring an HTTP request length longer than 110 to the number of total slots is more than 95%; Figs. 4 and 5).

This discussion indicates that the features shown in Table 4 (such as ASCII code "f") are effective for detecting malware infected because, in the case of infected traffic data, the appearance frequency of ASCII code included in domains is less than that of normal traffic data, and the HTTP request length is shorter than that of infected traffic data.

A.2.2 Trojan Horse

• Download malware in order to attack communications

After infection with a Trojan horse as a starting point, the Trojan horse downloads other malware from individual servers by using "HTTP GET" commands such as "*GET /vot.exe HTTP/1.0*" and "*GET /15Psv3zJ/4ah6NuS.exe HTTP/1.0.*" Most infected traffic communication is done using only "HTTP GET" since the amount of information in the payload is small. The number of breaks (¥r¥n) in infected traffic is fewer than that in normal traffic. The HTTP request length of infected traffic data is generally shorter than 20 while that of normal traffic data is generally longer than 110 (i.e., the ratio of the number of slots featuring an HTTP request length longer than 110 to the number of total slots is more than 95%; Figs. A·3 and A·4).

This discussion indicates that the features shown in Table 4 (ASCII code "e", etc.) are effective for detecting malware infection because, in the case of infected traffic data, the appearance frequency of ASCII code included in domains etc. is less than that of normal traffic data, and the HTTP request length is shorter than that of infected traffic data.

A.2.3 File-infected Virus

• C&C server connection by IRC communication (IRC connection)

A file-infected virus performs IRC communication and connects the target PC to a C&C server in preparation for performing infection activities. After the IRC communication is established, the file-infected virus performs infection activities such as downloading malware for communication attacks, DoS attacks against a target PC, etc. the file-infected virus performed infection activities such as downloading malware for attack communication in this analyzing. When performing IRC communication, specific strings (IRC domain: norks.org 001, etc.) repeatedly appeared with similarly high frequency in each slot in the communication contents. The appearance frequency of ASCII codes is typically variable in normal traffic data, but the appearance frequency of ASCII codes included in IRC communication (infected traffic data) was constant at 60 to 75.

• Download malware in order to attack communications

After infection with a file-infected virus as a starting point, the virus downloads other malware from individual servers by using "HTTP GET" commands such as "*GET /jiri.data HTTP/1.0*" and "*GET 44.data HTTP/1.0.*" The HTTP request length of infected traffic data is generally shorter than 110 while that of normal traffic data is generally longer than 110 (i.e., the ratio of the number of slots featuring an HTTP request length longer than 110 to the number of total slots is more than 95%; Figs. A·7 and A·8).

The appearance frequency of ASCII codes in IRC communication is greater than that of ASCII codes in normal traffic in each slot in terms of the communication contents and is a similar number in each slot. In other words, the appearance frequency of ASCII codes is variable in normal traffic data but that in IRC communication (infected traffic data) is constant at 60 to 75. Moreover, the HTTP request length is short in cases of infected traffic data. This indicates that the features shown in Table 4 (ASCII code "d", etc.) are effective for detecting malware infection.

In terms of the three types of malware, malware using HTTP communications performs "confirm Internet connection" and "download malware in order to perform attack communication" as infection activities. We identified the different behaviors (IRC connection and downloading from an outside server directly, etc.) for each type of malware when it was downloaded and performed infection activities such as attacking communications. In addition, we found that the string of payload information for downloading malware is different for each type of malware. We therefore conclude that the most effective feature for detecting infection depends on the type of malware.

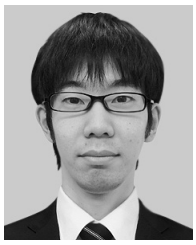
Editor's Recommendation

How to extract payload features form network traffic plays a critically important role when we want to detect unknown malware effectively and efficiently. In this paper, the authors comprehensively evaluate a wide range of feature extraction methods. In addition, the evaluation results are reliable based on the common MWS2012 (anti Malware engineering WorkShop 2012) Datasets. The implications carefully derived from the evaluation thus have a great impact on malware detection researches and practices.

(Chairman of SIGCSEC Kanta Matsuura)

Yusuke Otsuki received his B.E. degree in engineering from University of Electro-Communications, Tokyo, Japan in 2012. He is a master's degree student at the Graduate School of Electro-Communications, University of Electro-Communications.

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 rom Waseda University, Tokyo, Japan, in 2005 and 2008, respectively. Dr. Ichino is currently an assistant professor at the Graduate School of Informatics and Engineering, 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, IEEE.



Soichi Kimura received his B.E. degree in engineering from University of Electro-Communications, Tokyo, Japan in 2012. He is a master's degree student at the Graduate School of Electro-Communications, University of Electro-Communications.

Mitsuhiro Hatada was born in 1978. 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 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, 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.