Electronic Preprint for Journal of Information Processing Vol.21 No.4

Regular Paper

Mining Botnet Coordinated Attacks using *Apriori-PrefixSpan* Hybrid Algorithm

Masayuki Ohrui[1,a)] Hiroaki Kikuchi[2,3,b)] Nur Rohman Rosyid[4,c)] Masato Terada[5,d)]

Received: December 5, 2012, Accepted: June 14, 2013

Abstract: This paper aims to detect features of coordinated attacks by applying data mining techniques, namely *Apriori* with *PrefixSpan*, to the CCC DATAset 2008–2010, which comprises captured packet data and downloading logs. Data mining algorithms enable us to automate the detection of characteristics in large amounts of data, which conventional heuristics cannot deal with. *Apriori* achieves a high recall but with false positives, whereas *PrefixSpan* has high precision but low recall. We therefore propose a hybrid of these two algorithms. Our analysis shows a change in the behavior of malware over the past three years.

Keywords: botnet, data-mining, Apriori, PrefixSpan

1. Introduction

Malware has been evolving constantly. Conventional single-malware attacks have become less frequent any more recently, as shown by the number of malware downloading events observed by the Cyber Clean Center (CCC) [6] in **Fig. 1**, where the horizontal axis shows the weekly frequency of events.

However, instead of straightforward, single-malware attacks, the mainstream has shifted to more complicated, multiple-malware infections controlled within a botnet, with many servers infected by malware in a coordinated attack on a target host. In Ref. [5], there has been observed sequential infections in the CCC DATAset, containing packet data captured by 94 honeypots in which a honeypot is infected by multiple distinct malwares, scheduled in the same order. Although the servers were assigned different IP addresses, it turns out that there was a correlation between the malware infections. In this paper, we call such multiple infections made by several servers *coordinated botnet attacks*.

Understanding the coordinated botnet attacks are getting more important recently. Fist of all, the coordinated attack ensures high degree of robustness. Even if a server is detected and blocked, many other servers compensate the role of blocked server. Secondly, the information of coordinated attack patterns would help to improve the ability to respond. For instance, if we learn the frequency of downloading events for IP address and port numbers in the serious of download, we identify the most significant address or port to block, e.g., bottle-neck of botnet, which allows to minimize change of configuration. The statistics of coordinated pattern can be used for improvement of malware propagation models [10]. Finally, the analysis results in coordinated botnet attacks are useful for network forensics perspective. The particular sequence of attacks are used as a fingerprint of attacker, i.e., an evidence for unique identification to a botnet.

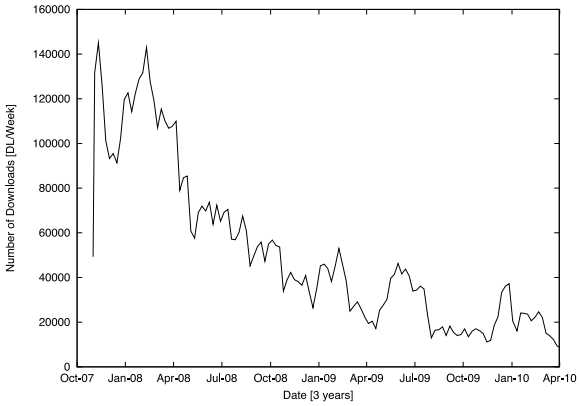

Fig. 1 Number of single malware downloads (2007–2010).

Moreover, in recent years, "Gumblar" and other Web-based malware have introduced an attack called *drive-by-download*, which involves many Web servers forcing target hosts to download malware, thereby increasing the resulting damage. It is almost impossible to trace the path of downloads manually because of the quantities and types of packets used in the drive-by-download attack. Instead, we need to use a data-mining algorithm for analysis.

There are two major data-mining techniques for extracting

1 Hitachi Ltd., Security & Smart ID Solutions Division
2 Department of Frontier Media Science, School of Interdisciplinary Mathematical Sciences, Meiji University, Nakano, Tokyo 164–8525, Japan
3 School of Information and Telecommunication Engineering, Tokai University, Minato, Tokyo 108–8619, Japan
4 School of Vocational, Universitas Gadjah Mada, Yogyakarta, Indonesia
5 Hitachi Ltd., Hitachi Incident Response Team (HIRT)
a) masayuki.orui.nx@hitachi.com
b) kikn@meiji.ac.jp
c) nrohmanr@ugm.ac.id
d) masato.terada.rd@hitachi.com

The primary version of this work has been published in the 13th and International Conference on Network-Based Information Systems (NBiS 2010), IEEE and the 14th and International Conference on Network-Based Information Systems (NBiS 2011), IEEE.

Table 1 Differences between Apriori and PrefixSpan.

	Apriori	PrefixSpan
Proponent	Agrawal et al. [1]	Pei et al. [2]
Extraction	Association rule	Sequential pattern
	(A, B → C)	(A, B, *, C)
Precision	Support, Confidence	Confidence
Feature	A set of items (unordered)	Sequence (in order)

valuable features of the malware from downloading logs, namely *Apriori* [1] and *PrefixSpan* [2]. The *Apriori* technique was designed to detect significant correlations within a set of items to extract rules for items with high support (a fraction of the subset of items). Support is a useful feature for detecting all possible coordinated behaviors among servers. However, since *Apriori* deals with a *subset of downloading events* without considering the order of events, it has a false-positive ratio. For example, a sequence of events *a* and then *b* is equivalent to one of *b* and *a* in *Apriori*. The detected coordination patterns in *Apriori* may contain false coordination such as two independent servers happening to work at almost the same time by chance. Therefore, its confidence is not high.

On the other hand, *PrefixSpan* considers the *sequence of downloading events* that was ignored in *Apriori*. It would therefore be expected to have a higher accuracy than *Apriori*. However, *PrefixSpan* does not evaluate the supporting of rules. Therefore, using the sequential pattern mining in *PrefixSpan* can improve the accuracy of the association rules by considering time series of downloading events that is the drawback of *Apriori* [4], [9]. **Table 1** summarizes the differences between *Apriori* and *PrefixSpan*.

In this paper, we examine these two data mining techniques, *Apriori* and *PrefixSpan*, in terms of a dataset of actual downloading events, namely the CCC DATAset 2008–2010 [5], [6]. The focus of our analysis is the changing behavior of malware over the past three years. Our experimental analysis investigates the features of and changes in coordinated attacks. Interestingly, the number of malware infections has been decreasing over these three years, suggesting to us that the mainstream of botnet attacks has shifted from single servers to multiple coordinated servers with Web-based drive-by-downloading malware.

Our Contribution

In summary, we make the following three contributions in this paper:

- We introduce a coordinated attack performed by botnet and show some properties of attack including the length of attack sequence (Section 4.3), the average duration of attack (Section 4.4), and the change of frequencies for three years (Section 4.5).
- We proposed a new hybrid scheme of Apriori and PrefixSpan for detection of coordinated botnet attacks, which improves accuracy of detection.
- We demonstrate our scheme to detect coordinated attacks from CCC Dataset. Our empirical analysis show that our hybrid scheme detects attacks accurately in practical use.

The rest of the paper is organized as follows. Section 2 describes the building blocks for this study, the two data-mining algorithms. In Section 3, we define the botnet coordinated attacks and show some typical behavior. In Section 4, we show the results of our analysis using the data-mining algorithms and accuracy of detection of coordinated attacks. The idea of hybrid is mentioned in Section 5 for improving the accuracy. Section 6 give the concluding remarks.

Table 2 A transaction example.

TID	A	B	C	D	E
1	1		1	1	
2		1	1		1
3	1	1	1		1
4		1			1

2. Building Blocks

2.1 The Apriori Algorithm

Apriori is a well-known algorithm for association-rule discovery described by Agrawal et al. [1]. It enables the efficient discovery of useful association rules by excluding rules whose support and confidence is smaller than given support and confidence thresholds. With the minimum-support condition, we can eliminate the examination of many useless candidate rules.

Association rules are of the form

$$X(antecedent) \Rightarrow Y(consequent)$$

from a given set.

Support is the probability that an association rule ($X \Rightarrow Y$) can be shown for a set of all transactions N, and is defined as

$$Supp(X \Rightarrow Y) = \frac{|X \cap Y|}{N}$$

Confidence is the probability that the rule is satisfied, namely the chance of Y being true if X is true. The definition is given by

$$Conf(X \Rightarrow Y) = \frac{|X \cap Y|}{|X|}$$

For example, the association rule $B, C \Rightarrow E$ in **Table 2** has support and confidence given by

$$Supp(B, C \Rightarrow E) = 2/4 = 0.5,$$
$$Conf(B, C \Rightarrow E) = 2/2 = 1.$$

Therefore, the rule $B, C \Rightarrow E$ has a support of 50% and a confidence of 100%. In other words, this rule occurs with a probability of 50%, and $B, C \Rightarrow E$ occurs with a probability of 100% whenever B and C occur.

We show Algorithm 1 [1] which generates candidate sets from a large item sets whose support is greater than $\epsilon \cdot |T|$. By C_k denote a candidate set for length k and $count[c]$ is a count of item set c in the dataset T.

2.2 The PrefixSpan Algorithm

Sequential pattern mining is a method for discovering subsequence patterns in a database of sequences, where each sequence comprises a list of elements and each element comprises a set of items. Given a user-specified minimum support threshold as a condition, sequential pattern mining aims to find all of the frequent subsequences, i.e., those subsequences whose occurrence frequency in the set of sequences is greater than or equal to the minimum support. The sequential-pattern-mining method called

Algorithm 1 Apriori

Input: dataset T and constant ϵ

```
1: L_1 = {1-item set}
2: k = 2
3: while L_{k-1} ≠ ∅ do
4:    C_k = {c ∈ a ∪ {b} | a ∈ L_{k-1}, b ∈ ⋃ L_{k-1}, b ∉ a}
5:    for transactions t ∈ T do
6:       C_t = {c | c ∈ C_k, c ⊂ t}
7:       for candidates c ∈ C_t do
8:          count[c] = count[c] + 1
9:       end for
10:   end for
11:   L_k = {c ∈ C_k | count[c] ≥ ε}
12:   k = k + 1.
13: end while
14: return L_2, ..., L_k.
```

Table 3 A sequence database.

Sequence id	Sequence					
100	PE	WO	TR			
200	PE	TR	WO			
300	BK	PE	TR	TS	WO	
400	TS	PE	PE	TR	WO	BK
500	PE	WO	TR	WO		

PrefixSpan (Prefix-projected Sequential pattern mining) was first proposed by Jien Pei [2].

Let a_i, b_j be items and let α_i, β_j be sequences of items such that $\alpha = \langle a_1 a_2 ... a_n \rangle$ and $\beta = \langle b_1 b_2 ... b_m \rangle$. Then α is **subsequence** of β, denoted by $\alpha \sqsubseteq \beta$, if and only if there exist integers $j_1, j_2, ..., j_n$, where $1 \leq j_1 < j_2 < ... < j_n \leq m$, such that $a_1 = b_{j_1}, a_2 = b_{j_2}, ..., a_n = b_{j_n}$. A **sequence database** S is a set of tuples $\langle sid, s \rangle$, where sid is a **sequence_id** and s is a **sequence**. The support of a sequence α in a database S is the number of tuples in the database containing α, i.e., $support(\alpha) = |\{\langle sid, s \rangle | \langle sid, s \rangle \in S, \alpha \sqsubseteq s\}|$. Given a positive integer min_sup as a support threshold, a sequence α is called a **frequent sequential pattern** in database S if the sequence is contained in at least min_sup tuples in the database, i.e., $support(\alpha) \geq min\_sup$. The number of items in a sequence is called the **length** of the sequence, and a sequential pattern of length ℓ is called an **ℓ-pattern**.

In terms of the *PrefixSpan* algorithm, let α and β be sequences $\langle a_1 ... a_n \rangle$ and $\langle b_1 ... b_m \rangle$, respectively.

(1) **Prefix** and **Postfix**: sequence α is a prefix of β if and only if $a_i = b_i$ for $i = 1, ..., m$. For example, $\langle a\ a\ b\ c \rangle$ is a prefix of $\langle \boldsymbol{a\ a\ b\ c}\ d\ d\ a\ b \rangle$, with the sequence after the prefix being the postfix. That is, $\langle d\ d\ a\ b \rangle$ is the postfix in $\langle a\ a\ b\ c\ \boldsymbol{d\ d\ a\ b} \rangle$.

(2) **Projection**: Let α, β, γ be sequences such that $\beta \sqsubseteq \alpha, \gamma \sqsubseteq \alpha$. Sequence γ is β-**projection** of α if and only if (1) β is prefix of γ, and (2) there exists no longer subsequence of α such that β is its prefix. For example, if $\alpha = \langle a\ a\ b\ c\ d\ c\ d\ a\ b \rangle$ and $\beta = \langle a\ a\ b\ c \rangle$, then β-projection of α is $\gamma = \langle d\ c\ d\ a\ b \rangle$.

As an example, given the sequence database S in **Table 3** and a user-specified $min\_sup = 2$, the sequential patterns in S can be mined by the *PrefixSpan* method using the following three steps:

Step 1: Find 1-pattern sequences.

Scan the database S once to discover all frequent items in the sequences. These are ⟨PE⟩:5, ⟨WO⟩:5, ⟨TR⟩:5, ⟨BK⟩:2 and ⟨TS⟩:2, where ⟨pattern⟩:*count* is the pair of the pattern and support count.

Step 2: Distribute the search space.

The projected database can be distributed into the following five subsets according to the five prefixes which resulted from Step 1: (1) those having prefix ⟨PE⟩;...; and (5) those having prefix ⟨TS⟩.

Step 3: Find the subsets of sequential patterns.

These can be mined by constructing the corresponding *projected databases* and exploring each recursively.

3. Coordinated Botnet Attacks

3.1 Definition

Botnets are able to mount coordinated attacks by multiple servers to infect a victim with a set of malwares [7]. As we have mentioned in Introduction, **Table 5** shows sequential infections observed the CCC DATAset 2009, in terms of data packets captured by 94 honeypots [5], in which a honeypot is infected by the three malwares PE_VIRUT.AV, TROJ_BUZUS.AGB and WORM_SWTYMLAI.CD scheduled in the same order. In Table 5, we note that PE_VIRUT.AV remains highly ranked for 3 years and that PE_VIRUT.AV is the malware that starts the coordinated attacks. In addition, the PE family is the most common, although the number of infections decreases over the three years.

The series of malware download is called coordinated attack, as follows.

Definition 3.1 A *coordinated botnet attack* (or a coordinated attack) is an series of malware infections to a victim host from multiple server controlled by a botnet.

We simply call a *botnet* without regards to size or organization that the attacker used for control some servers to deliver malware to victim.

3.2 Evidence of Coordinated Attack

Any attack from botnet can be classified as the coordinated attack. However, the opposite, i.e., series of malware download is not always made by botnet. The observed sequence may be possible but not efficient to prove the set of malware are delivered under control of the same attacker. To the sake of argument, we show a hypothesis testing. Suppose that the set of malware downloads were random.

According to the frequencies of malware downloading events in **Table 4**, the probability of given malware to be infected is

$$Pr[\texttt{PE\_VIRUT.AV}] = \frac{\text{frequency}}{\text{total downloads}} = \frac{700}{2357} = 0.30.$$

Similarly, $Pr[$ TROJ_BUZUS.AGB $] = 0.06$ and $Pr[$ WORM_SWTYMLAI.CD $] = 0.08$. Under our hypothesis, a probability that three events happen in this order is $0.30 \cdot 0.06 \cdot 0.08 = 0.00144$. The probability of the sequence to be observed for three times a month is 2.70×10^{-9}, which is too small. Therefore, we reject the hypothesis of random events and hence claim the sequence of malware was made by servers under a control of the same attacker, referred as *coordinated botnet attacks*.

Table 5 Examples of coordinated malware attacks.

Time	Source IP address	Dst Port	Protocol	MW
0:02:11	124.86.***.111	47556	TCP	PE_VIRUT.AV
0:03:48	67.215.*.206	80	TCP	TROJ_BUZUS.AGB
0:03:48	72.10.***.195	80	TCP	WORM_SWTYMLAI.CD
0:36:46	124.86.**.109	33258	TCP	PE_VIRUT.AV
0:36:52	72.10.***.195	80	TCP	WORM_SWTYMLAI.CD
0:36:52	67.215.*.206	80	TCP	TROJ_BUZUS.AGB
0:46:56	124.86.**.109	33258	TCP	PE_VIRUT.AV
0:48:52	67.215.*.206	80	TCP	TROJ_BUZUS.AGB
0:48:52	72.10.***.195	80	TCP	WORM_SWTYMLAI.CD

Table 4 Top 5 malware observed in March 2009.

No.	Malware	Freq.	Probability
1	PE_VIRUT.AV	700	0.296988
2	UNKNOWN	240	0.101824
3	WORM_SWTYMLAI.CD	198	0.084005
4	BKDR_POEBOT.GN	166	0.070429
5	TROJ_BUZUS.AGB	146	0.061943
	⋮		
68	BKDR_RBOT.PA	1	0.000424
Sum		2357	1

3.3 Complexity in Detecting Coordinated Attack

Detecting coordinated attack is not easy. Let n be a number of distinct malware in a dataset. There are $\binom{n}{2}$ pairs of malware to be 2-item association rules. For a pair (A, B), there are two $A \Rightarrow B$ and $B \Rightarrow A$ rules. Hence, the total number of 2-item rules is $\binom{n}{2} \cdot 2$. If we are given a k-item set, defined as C_k in Algorithm 1, there are $2^k - 2$ variations, where 2 corresponds $\emptyset \Rightarrow C_k$ and $C_k \Rightarrow \emptyset$. Consequently, taking summation for all item sets, we have the total number of association rules as

$$\sum_{k=2}^{n} \binom{n}{k} \cdot (2^k - 2) \tag{1}$$

In Table 4, we find $n = 68$ kinds of malware for one month and the total of possible rules is $278128389443103215446927070580050 = 2.7 \times 10^{32}$, which is not tractable in space or in time. Therefore, we need an efficient algorithm to detect association rules.

3.4 Experimental Data

The CCC DATAset contains the access logs for attacks over the three years from November 1, 2007 to April 30, 2010, shown in **Table 6**. The 94 independent honeypots were used to observe malware download performed in the Japanese tier-1 backbone under the coordination of the CCC. The honeypots are rebooted periodically, at 20-minute intervals. We call these time intervals *time slots* throughout this paper. A day's observation yields 72 time slots. A transaction is the list of malware names that are downloaded in one time slot. Similarly, the malware downloading logs are divided in terms of time slots.

The malware names in CCC Dataset were exactly identified based on the latest signature file maintained by Trend Micro [5]. The signature file were incrementally updated while the honeypots were observing network. The malware failed to be identified is specified explicitly as "UNKNOWN."

Table 7 summaries the specific environment to collect malware downloading events.

Table 6 CCC DATAset statistics.

CCC DATAset	# of honeypot	duration	# of records
2009	94	2007/11/1 – 2008/4/30	2,942,221
2010	92	2008/5/1 – 2009/4/30	1,162,093
2011	72	2010/5/1 – 2011/1/31	158,734

Table 7 Specification of environment to collect malware.

attribute	value
guest OS	Windows 2000, Windows XP SP1
signature	Trend Micro
attribute	time, source/destination IP address, source/destination port, hash value of malware, malware name, file name
format	CSV

4. Analysis

4.1 Experiment

4.1.1 Analysis Objectives

Our analysis is motivated by the following questions;

(1) What combinations of malware are frequently used for the botnet coordinated attack?
The botnet may use the set of malware to perform the coordinated attack. If we learn the frequent pattern, we can identify the bottleneck of the patterns and use the information to prevent the further attack.

(2) How uniformly are the coordinately attacks performed?
If the botnet performed the attack to randomly chosen address, all our honeypots observed the attack uniformly. If not, we may study botnet's destination selection strategy and can use it to prevent further attack.

(3) How long does the botnet perform a single coordinated attack?
The botnet are supposed to be quite adaptive against the defense and frequently change their active servers. If we find the duration of a single particular coordinated attack, the fundamental statistics of botnet attack can be clarified.

(4) How accurate does the data-mining algorithm detect the coordinate attack?
Both of Apriori and PrefixSpan are useful algorithm but we are not sure which is better for our purpose.

In order to answer the above questions, we have the following experiments. In the parenthesis, we specify the corresponding subsequent sections for showing the experimental results.

(1) Observed the malware download using honeypots and use the signature to detect the corresponding malware names (CCC Dataset).

(2) Apply Apriori to the log file of malware names and have the

association rules of malware (Section 4.2).
(3) Apply Apriori to the log file of downloading honeypot IDs and have the association rules of honeypot (Section 4.3).
(4) Use the top association rules of malware to find the duration of attack (Section 4.4).
(5) Apply PrefixSpan to the log file of malware names and compare with the Apriori in terms of the accuracy (Section 4.5).

4.1.2 Input Data

Table 8 shows the sample of input file to Step 2 (Apriori) and 5 (PrefixSpan). The detected malware names are listed in the order of detection in each of time slots. These names are used as item set in data-mining algorithm described in Section 2.

We observe some names labeled as UNKNOWN, which means fault of identification from the signature database. Since the data-mining process discloses the most frequent items only, we can omit these fault of identification without depending on our results.

4.1.3 Parameter Optimization

Before performs data-mining algorithms, we need to specify parameters of mining, i.e., min_sup, min_conf. The choice of these parameters are heuristically fixed and tuned, as designed in these algorithms. If the min_sup is too high, no frequent items sets is given. Then, we pick a lower value for min_sup and try again. This seems to be inefficiently, but in fact the results are obtained very quickly and hence we can do this very practically.

4.2 Association Rules for Malware

Coordinated malicious servers send the same type of malware to a single target host. **Table 9** shows an example of malware sequences observed for each time slot, showing 58 infected slots out of 145 slots. The most frequent infection involves 11 distinct malware names in a single slot.

We applied the Apriori algorithm to the dataset of malware shown in Table 9 and successfully discovered significant association rules for the malware shown in **Table 10**. This result identifies all association rules with support above 10% and confidence above 80%.

We are interested in whether the pattern PE_VIRUT.AV ⇒ TROJ_BUZUS.AGB, WORM_SWTYMLAI.CD shown in Table 10 is automatically detected or not. Unfortunately, Table 10 does not contain exactly this rule, but has several similar association rules, such as Rule No.5: PE_VIRUT.AV, TROJ_BUZUS.AGB ⇒ WORM_SWTYMLAI.CD and Rule No.6: PE_VIRUT.AV, WORM_SWTYMLAI.CD ⇒ TROJ_BUZUS.AGB. From the observations in Table 10, we found a significant correlation between TROJ_BUZUS.AGB and WORM_SWTYMLAI.CD in Rule Nos. 1, 2 and 3.

Table 8 Sample input data transformed from CCC dataset.

time slot	malware names
0	TROJ_SYSTEMHI.BQ
1	KDR_AGENT.ANHZ UNKNOWN TROJ_SYSTEMHI.BQ UNKNOWN
2	PE_BOBAX.AH
3	PE_BOBAX.AH UNKNOWN BKDR_AGENT.ANHZ
⋮	⋮
15323	PE_VIRUT.AV TROJ_IRCBRUTE.BW WORM_AUTORUN.CZU

4.3 Dependency on Honeypots

We are interested in the degree to which extracted association rules depend on particular honeypots. We show the numbers of honeypots that have observed the top 10 association rules of malware in **Table 11**.

We investigated 94 honeypot IDs during March 13, 2009. For example, we found that 36 of the 94 honeypots observed common Rule No.1, suppressing differences of support and confidence.

Figure 2 shows the number of association rules in terms of the number of distinct honeypots that observe the rule, denoted by k. The vertical axis shows the number of distinct association rules $N(k)$, where k honeypots detect the rule. We note that the most common association rules using TROJ_BUZUS.AGB and WORM_SWTYMLAI.CD are observed by 36 honeypots. These widely observed honeypots can be considered as those being used for coordinated attacks. We also note that only specific malware is used to coordinate the attacks.

4.4 Lifecycle of Association Rules for Malware

The duration of coordinated attacks is short. **Figure 3** shows the distribution of activity for the top three association rules ex-

Table 9 Sequences of malware observed in a time slot.

Time Slot	Sequence of Malware			
0	PE_VIRUT.AV	TROJ_BUZUS.AGB	WORM_SWTYMLAI.CD	
2	WORM_ALLAPLE.IK	PE_VIRUT.AV	WORM_SWTYMLAI.CD	TROJ_BUZUS.AGB
3	PE_VIRUT.AV	TROJ_BUZUS.AGB	WORM_SWTYMLAI.CD	PE_VIRUT.AV
14	BKDR_POEBOT.GN	TROJ_BUZUS.AGB	WORM_SWTYMLAI.CD	
15	BKDR_MYBOT.AH	PE_VIRUT.AV		
⋮				
141	PE_BOBAX.AK	WORM_SWTYMLAI.CD	WORM_AUTORUN.CZU	WORM_IRCBOT.CHZ

Table 10 Association rules for malware infection.

Rule.	Antecedent			Consequent	Supp	Conf
1		TROJ_BUZUS.AGB	⇒	WORM_SWTYMLAI.CD	41.4	100
2		WORM_SWTYMLAI.CD	⇒	TROJ_BUZUS.AGB	41.4	88.9
3	TROJ_BUZUS.AGB	BKDR_POEBOT.GN	⇒	WORM_SWTYMLAI.CD	10.3	100
4	WORM_SWTYMLAI.CD	BKDR_POEBOT.GN	⇒	TROJ_BUZUS.AGB	10.3	100
5	PE_VIRUT.AV	TROJ_BUZUS.AGB	⇒	WORM_SWTYMLAI.CD	29.3	100
6	PE_VIRUT.AV	WORM_SWTYMLAI.CD	⇒	TROJ_BUZUS.AGB	29.3	100
*	PE_VIRUT.AV	⇒ WORM_SWTYMLAI.CD		TROJ_BUZUS.AGB	N/A	N/A

Table 11 Number of honeypots having association rules for malware on March 13, 2009.

No.	Antecedent			Consequent	Honeypots
1		TROJ_BUZUS.AGB	⇒	WORM_SWTYMLAI.CD	36
2		WORM_SWTYMLAI.CD	⇒	TROJ_BUZUS.AGB	36
3	TROJ_BUZUS.AGB	BKDR_VANBOT.AHH	⇒	WORM_SWTYMLAI.CD	12
4	WORM_SWTYMLAI.CD	BKDR_VANBOT.AHH	⇒	TROJ_BUZUS.AGB	12
5		TROJ_DLOADR.CBK	⇒	UNKNOWN	8
6	TROJ_BUZUS.AGB	PE_VIRUT.AV	⇒	WORM_SWTYMLAI.CD	7
7	WORM_SWTYMLAI.CD	PE_VIRUT.AV	⇒	TROJ_BUZUS.AGB	7
8	PE_VIRUT.AV	TROJ_BUZUS.AGB	⇒	WORM_SWTYMLAI.CD	6
9		TROJ_AGENT.ANDF	⇒	UNKNOWN	6
10	PE_VIRUT.AV	WORM_SWTYMLAI.CD	⇒	TROJ_BUZUS.AGB	6

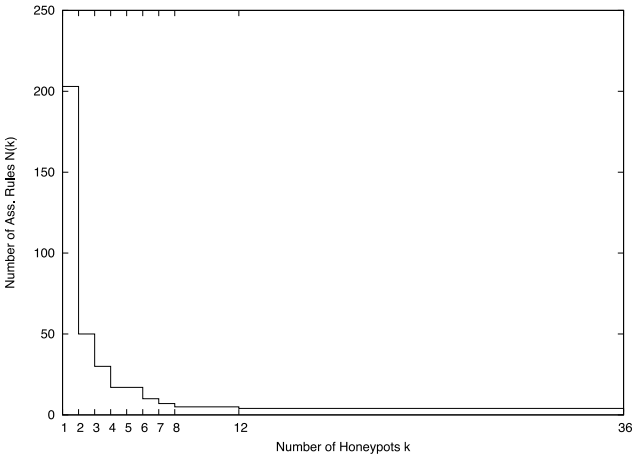

Fig. 2 Number of association rules in terms of the number of honeypots observing the rules.

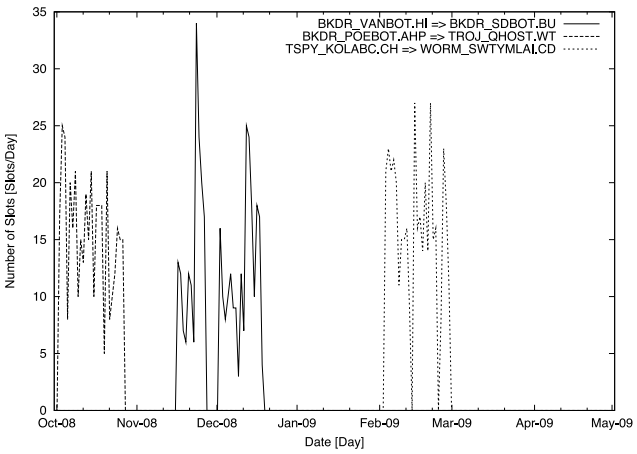

Fig. 3 Distribution of activity for the top three rules (omitting UNKNOWN).

cept for UNKNOWN, defined by:

(1) BKDR_VANBOT.HI ⇒ BKDR_SDBOT.BU

(2) BKDR_POEBOT.AHP ⇒ TROJ_QHOST.WT

(3) TSPY_KOLABC.CH ⇒ WORM_SWTYMLAI.CD.

The horizontal axis shows the date of observation, and the vertical axis shows the number of slots in which the top three association rules were observed. The average duration for three rules is 26.3 days. The reason why the duration is short is that a short period of coordinated attacks is hard to detect. Moreover, the coordinated pattern is constantly renewed every time new malware is developed.

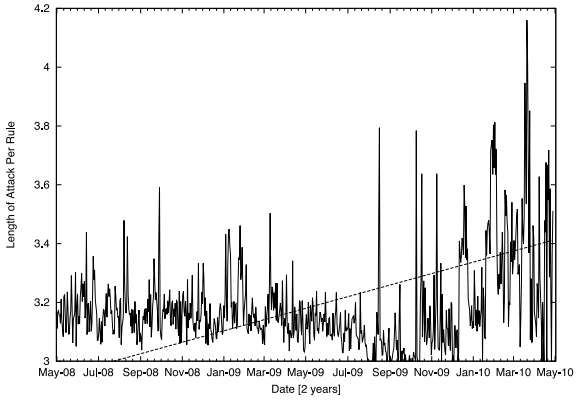

Fig. 4 Average length of coordinated attacks per rule.

4.5 Changes in Coordinated Attacks

We investigated the number of types of malware used to perform coordinated attacks. For this purpose, we applied the PrefixSpan algorithm, which can distinguish patterns with different infection ordering, aiming to extract the coordinated infection patterns for all honeypots.

Figure 4 illustrates the change in the average number of types of malware used in attacks. The vertical axis shows the length of attack per rule, i.e., the number of distinct malware consisting of coordinated attack, with regard to the date of observation. The number of types of malware increases, despite the decrease in the overall number of attacks. We stress that this shows that coordinated attacks are becoming more complex and advanced than previously. For example, the malware downloaded with HTTP GET, which was used by two malwares in 2008 and 2009, was observed five times in malware in 2010. We therefore conclude that coordinated attacks are clearly complicated.

5. The Hybrid Approach using Apriori and PrefixSpan

5.1 Comparison between Apriori and PrefixSpan

We evaluated the two automated algorithms, Apriori and PrefixSpan, in terms of their accuracy in detecting coordinated malware attacks.

We show the sample output of Apriori and PrefixSpan for the log data of February 4th, 2009, in **Figs. 5** and **6**, respectively. In Apriori, the recode is of the form ($|X|$, confidence)[*1] and minimum support is 5 and minimum confidence 80 % was used. In PrefixSpan, the support is indicated at the end of line. We investigate the underlined rules for both experimental results and show

[*1] The support is derived from $|X|$ by $|X| \cdot \text{confidence}/N = |X \cap Y|/N$.

```
PE_VIRUT.AV, BKDR_POEBOT.GN ⇒ TSPY_KOLABC.CH (8, 87.5)
PE_VIRUT.AV, BKDR_POEBOT.GN ⇒ WORM_SWTYMLAI.CD (8, 100.0)
PE_VIRUT.AV, WORM_SWTYMLAI.CD ⇒ BKDR_POEBOT.GN (10, 80.0)
PE_VIRUT.AV, TSPY_KOLABC.CH ⇒ WORM_SWTYMLAI.CD (9, 100.0)
PE_VIRUT.AV, WORM_SWTYMLAI.CD ⇒ TSPY_KOLABC.CH (10, 90.0)
BKDR_POEBOT.GN, TSPY_KOLABC.CH ⇒ WORM_SWTYMLAI.CD (14, 100.0)
BKDR_POEBOT.GN, WORM_SWTYMLAI.CD ⇒ TSPY_KOLABC.CH (16, 87.5)
TSPY_KOLABC.CH, WORM_SWTYMLAI.CD ⇒ BKDR_POEBOT.GN (17, 82.4)
PE_VIRUT.AV, BKDR_POEBOT.GN, TSPY_KOLABC.CH ⇒ WORM_SWTYMLAI.CD (7, 100.0)
PE_VIRUT.AV, BKDR_POEBOT.GN, WORM_SWTYMLAI.CD ⇒ TSPY_KOLABC.CH (8, 87.5)
```

Fig. 5 The sample output of Apriori.

Table 12 Comparison between Apriori and PrefixSpan.

Date	Apriori			PrefixSpan		
	Rule	Slots	True [Slots]	Rule	Ptns	True [Ptns]
2009/02/03	WORM, BKDR ⇒ TSPY	4	4	TSPY ⇒ WORM ⇒ TKDR	3	9
2009/02/04	BKDR, TSPY ⇒ WORM	14	14	TSPY ⇒ BKDR ⇒ WORM	3	29
				TSPY ⇒ WORM ⇒ BKDR	7	
				WORM ⇒ BKDR ⇒ TSPY	4	
				WORM ⇒ TSPY ⇒ BKDR	12	
			⋮			
2009/02/28	BKDR, TSPY ⇒ WORM	7	7	TSPY ⇒ WORM ⇒ BKDR	5	14
	BKDR, WORM ⇒ TSPY	7		WORM ⇒ TSPY ⇒ BKDR	3	
Sum		464	315		482	575
	false positive (464 > 315)			false negative (482 < 575)		

```
PE_VIRUT.AV TSPY_KOLABC.CH WORM_SWTYMLAI.CD 4
TSPY_KOLABC.CH BKDR_POEBOT.GN TSPY_KOLABC.CH 4
WORM_SWTYMLAI.CD BKDR_POEBOT.GN TSPY_KOLABC.CH 4
WORM_SWTYMLAI.CD TSPY_KOLABC.CH PE_VIRUT.AV 4
WORM_SWTYMLAI.CD TSPY_KOLABC.CH WORM_SWTYMLAI.CD 4
WORM_SWTYMLAI.CD WORM_SWTYMLAI.CD BKDR_POEBOT.GN 5
PE_VIRUT.AV WORM_SWTYMLAI.CD TSPY_KOLABC.CH 6
PE_VIRUT.AV TSPY_KOLABC.CH BKDR_POEBOT.GN 7
PE_VIRUT.AV WORM_SWTYMLAI.CD BKDR_POEBOT.GN 7
TSPY_KOLABC.CH WORM_SWTYMLAI.CD BKDR_POEBOT.GN 7
WORM_SWTYMLAI.CD TSPY_KOLABC.CH BKDR_POEBOT.GN 12
```

Fig. 6 The sample output of PrefixSpan.

Table 13 Accuracy in Apriori.

	Coordinated	Non-Coordinated	Sum
Extracted	315	149	464
Non-Extracted	0	N/A	N/A
Sum	315	149	464

Table 14 Accuracy in PrefixSpan.

	Coordinated	Non-Coordinated	Sum
Extracted	482	0	482
Non-Extracted	93	N/A	93
Sum	575	N/A	575

Table 15 Recall and precision.

	Apriori	PrefixSpan
Recall	315/315 = 1	482/575 = 0.838
Precision	315/464 = 0.678	482/482 = 1

the comparison in **Table 12**.

Our target coordinated attack to be detected by these algorithms was the sequence of malware: TSPY_KOLABC.CH, WORM_SWTYMLAI.CD and BKDR_POEBOT.GN that had been reported by Trend Micro [8]. The accuracy of Apriori is given as the frequency of detected time slots, indicated in columns labeled as "Slots," as a proportion of the true time slots identified by manual investigation. The accuracy of PrefixSpan is defined as the proportion of detected coordinated-attack patterns within the true patterns, and labeled as "Ptns" in the table.

For example, Apriori extracts all four of the coordinated attacks on 3rd February. PrefixSpan detects three correct patterns, missing six patterns out of nine, on the same day. On 28th February, Apriori made false detections in seven slots. The reason for these false positives is that Apriori considers all possible combinations of malware, without observing the order of detection. On the other hand, PrefixSpan had a relatively low false-positive rate compared with Apriori, although it had false negatives. For example, on 4th February, (number of true patterns) − (sum of detected patterns) = $29-(3+7+4+12)=3$, which implies there were 3 missing patterns at too low a frequency.

Consequently, Apriori is good at detecting those time slots when coordinated attacks may have occurred, whereas PrefixSpan is useful for detecting exact coordinated patterns of malware. We can combine these two automated approaches for the accurate detection of attacks.

5.2 Accuracy in Detection

Our comprehensive investigation of the CCC DATAset is summarized in **Tables 13** and **14**, with respect to the accuracy of Apriori and PrefixSpan, respectively. Note that Apriori aims to detect coordinated time slots and PrefixSpan detects sequence patterns in malware. The tables shows that Apriori has 149 false positives (slots) out of 464 and no false negatives, whereas PrefixSpan has no false positives (patterns) but fails to detect 93 patterns out of 575. To summarize this, we use two criteria, *precision*, defined as the fraction of correctly detected slots (patterns) in all detected slots, and *recall*, defined as the fraction of correctly detected slots (patterns) in all slots with attacks. These are shown in **Table 15**. Apriori achieves high recall but with false positives. PrefixSpan can be tuned, using an appropriate minimum support bound, to

Table 16 Accuracy of Hybrid approach for three datasets.

DATAset	Date	Accuracy	Apriori	PrefixSpan	Hybrid
2009	2009/02	Recall	315/315 = 1	482/575 = 0.838	545/575 = 0.947
		Precision	315/464 = 0.678	482/482 = 1	482/482 = 1
2010	2009/12	Recall	251/251 = 1	535/592 = 0.903	575/592 = 0.971
		Precision	251/395 = 0.639	535/535 = 1	575/575 = 1
2011	2010/06	Recall	51/51 = 1	42/94 = 0.447	74/94 = 0.787
		Precision	51/114 = 0.447	42/42 = 1	74/74 = 1
Average		Recall	1	0.729	0.902
		Precision	0.588	1	1
Std. Deviation		Recall	0	0.201	0.082
		Precision	0.101	0	0

Algorithm 2 Hybrid with Apriori and PrefixSpan

input: dataset T

Step 1. Apply Apriori algorithm to T and get frequent item sets $L_2, \ldots, L_k$.

Step 2. Apply PrefixSpan algorithm to $L_2, \ldots, L_k$ with a certain minimal support.

Output. All significant association rules, corresponding to coordinated attacks.

filter out useless patterns.

5.3 A Hybrid Approach with Apriori and PrefixSpan

From our observation, we come up with idea of hybridizing Apriori and PrefixSpan. We first apply Apriori to detect time slots containing potential coordinated attacks because we have no a priori knowledge about likely correlations between malware. After Apriori has identified possible slots, we apply the PrefixSpan algorithm to improve the accuracy. For example, on February 4th, Apriori and PrefixSpan detected nine patterns and 32 patterns [*2], respectively. However, after Apriori detected three major malwares, `TSPY`, `WORM` and `BKDR`, the second filter of PrefixSpan reduced the number of false alerts from 32 to four patterns. The final four patterns that begin with `TSPY` and `WORM` are listed in Table 12, and labeled as "PrefixSpan." The results suggest that these four patterns are the most likely sequences of malware used in a botnet. For simplicity, we will concentrate on the three malwares of interest in this example. In practice, we would deal with many unrelated malwares observed in the same period of time.

We summarize our hybrid approach with Algorithm 2. The proposed scheme is a simple cascade of two independent schemes and hence contains some redundant procedures such as finding frequent item sets called by both schemes. If we skip the common procedures, the performance of the scheme can be improved. We leave it as one of future study.

The hybrid approach improves detection accuracy as shown in **Table 16**, where we test the hybrid approach for three datasets, CCC DATAset 2009, 2010, and 2011 [5], [6] in terms of recall and precision. With hybrid approach, the recall of PrefixSpan is improved from 0.838 to 0.947 in CCC DATAset 2009, while the precision of Apriori increases from 0.678 to 1.0. Similar improvements are observed in DATAset 2010 and 2011. The average recall is 0.902 with confidence interval of 95 % of $\pm 2\sigma = \pm 0.164$, which follows $0.902 - 0.164 = 0.738 > 0.729$. The average precision always greater than that of Apriori with significant difference. Hence, we conclude that the hybrid approach improves accuracy of Apriori and PrefixSpan with high confidence.

[*2] The detected 9 and 32 patterns are not shown in Table 12.

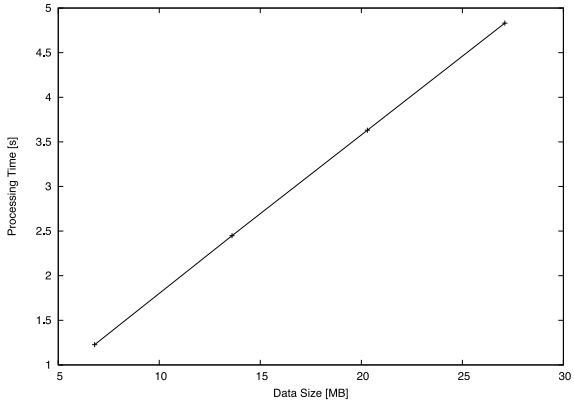

Fig. 7 Processing time for Apriori with regard to data size.

Note that the recall of Apriori, 1, is better than that of hybrid scheme, 0.902 in Table 16. The hybrid does not always improve both recall and precision. However, the improvement from PrefixSpan, which has recall of 0.729, must be more significant because the hybrid is free from false rules. Further improvement is one of our future studies.

5.4 Limitation of Proposed Method in terms of Performance

The scalability of the proposed method depends on that of two primitive data mining algorithm.

Figure 7 shows the processing time of Apriori with regards to the input data size, varying duration of observation in CCC Dataset 2009 from 3 month to 12 month, corresponding to 6.7 MB to 27.1 MB in size. It demonstrates the Apriori runs in linear time with input size. The internal analysis reveals the reading file is the bottleneck of performance as far as the meaningful support and confidence are given. Although hybrid approach takes double in time, the total time to detect all coordinated attacks is expected to be less than 10 seconds. If we require the analysis to be done within in a minute, we say the limitation of application is about 10 years. Therefore, we conclude that the proposed method scales well.

5.5 Case Study in Proposed Scheme

We illustrate the advantage of our proposed method with case studies.

(1) Library with Wi-Fi network [11]. The Rotterdam Library, Netherland, has about 1,000 visitors per day, mostly students, connect their devices to the library's free Wi-Fi network, which recurs more than 4,000 pieces of malware in-

Table 17 Comparison of detection studies using data-mining algorithms.

schemes	Ref. [13]	Ref. [14]	Ref. [16]	Ref. [15]	Ref. [19]	Ref. [18]	Proposed
target	malware	malware	malware	malware	network	network	network
items	read/write	APIs sequence	kernel-events	APIs	IP, port	IP, TCP	IP, Malware name
dataset	synthesized	30,000 samples	samples	samples	CCC Dataset	backbone	CCC Dataset
mining	AprioriAll	OOA-FP	MAFIA and PrefixSpan	PrefixSpan*	N/A	Apriori and KL	Apriori and PrefixSpan

fection on the public access terminals. Most existing anti-virus, anti-spam, IDP did not work against advanced persistent threats with many kinds of malware.

Our proposed scheme is potentially used to detect coordinated attacks and hence altering any outbound packets controlled by infected hosts. The list of malware, Table 11, could help for identifying significant malware to be detected.

(2) University Network [12]. The university of Baltimore expands significant manual resources to keep up with blacklisting IP addresses and clean up malware infection. Despite ongoing education efforts, users would continue to respond phishing and visit hostile website.

Our scheme allows to automate the blacklisting processes with hybrid two data mining algorithm. The blacklisted address should be marked as malicious for an average duration, such as Fig. 4.

5.6 Related Works

There are many studies for detection technology using the data-mining algorithms including Apriori and PrefixSpan. These studies are classified into two categories; malware detection and intrusion detection.

In the first category of malware detection, the algorithm can be used to extract the typical sequence of APIs captures, memory/storage accesses invoked by a given malware. In Ref. [13], Hu and Panda proposed a data-mining algorithm for intrusion detection. They had experiment on synthetic database for malicious transaction, consisting of read and write sequences. They used the AprioriAll for generating sequential patterns. Ye, Wang, Li and Ye had developed the malware detection system, called, IMDS, in Ref. [14]. The IMDS analyzes Windows API execution sequences called by PE files and uses an object-oriented association mining algorithm, OOA-FP, instead by Apriori. With about 30,000 samples of malware, they evaluated the accuracy of the proposed system and showed the comparison to the well-known data-mining algorithms. LaRosa, Xiong, and Mandelberg presented a framework for mining kernel trace data to detect interesting inter-process communication patterns and runtime execution pattern in operating system trace logs in Ref. [16]. They used the Linux Trace Toolkit for collecting the kernel-events, e.g., open, alloc, syscall, for file system, and memory. They combined the frequent item-sets algorithm, MAFIA [17], and the sequence mining algorithm, similar to our proposed scheme. In Ref. [15], Wang, Tan, Pan and Xi proposed a behavior-based detection system in which the PrefixSpan* algorithm mines association rules from malicious code samples. They shows some lengh-2 patters from malware including 7 virus, 7 Trojans, 6 worms. They combined the expert system with the pattern mining algorithm.

The second category is the network-based intrusion detection. Automated approach is highly useful here since the volumes of flows observed from networks are too large to investigate by human. In Ref. [18], Brauckhoff, Dimitropoulos and Wagner proposed an anomaly detection algorithm using metadata based on the histogram of features, e.g., protocol, IP address, port, TCP flags, flow size, packet size, flow duration. Chosen significant features, they used the Apriori algorithm to detect the frequent item sets and then tested the Kullback-Leibler distance for anomaly detection. In Ref. [19], Takemori, Fujinaga, Sayama, and Nishigaki had developed a system, named as "Botnet visualizer" for identifying link between the C&C servers and the compromised hosts. For evaluation, they used the CCC Dataset [5], but did not use any data-mining algorithm for their purpose. Other studies based on the CCC Dataset can be found in Refs. [3], [4], [7], [9].

5.7 Comparison between the Proposed schema and the Related Works

Table 17 shows the comparison between the proposed scheme and the related works in terms of target of detection, the items to be mined, and the algorithms used to analyze.

6. Conclusions

We have reported on the characteristics and evolution of coordinated attacks, using data from the CCC DATAset for the past three years. Despite the number of coordinated attacks having decreased, the number of distinct malwares used in coordinated attacks has increased.

Our experiment successfully extracts the particular association rules of malware frequently observed by multiple honeypots. The malware servers are widely distributed in the internet and there is few servers used to send malware to many target hosts. The duration of coordinated attacks were very short. This implies that the coordinated attacks are controlled by a botmaster that regulates the ratio of attacks and stops the attack before it is detected. The data-mining algorithms, Apriori and PrefixSpan, are useful to detect the features of botnet attacks and to predict the future events. The Apriori has 149 false positive out of 464 slots (32.1%) and no false negative, whereas PrefixSpan has no false positives but fails to detect 93 patterns out of 575 (16.2%). Hence, combining these two algorithms, i.e., after Apriori has identified possible slots, we apply the PrefixSpan to improve the accuracy. The automated combination of Apriori and PrefixSpan is one of our future studies.

References

[1] Agrawal, R., Imielinski, T. and Swami, A.: Mining Association Rules between Sets of Items in Large Databases, *Proc. ACM SIGMOD-93*, pp.207–216 (1993).

[2] Pei, J., Han, J., Behzad, M.A. and Pinto, H.: PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth, *Proc. 17th Int'l Conf. on Data Engineering*, pp.215–224 (2001).
[3] Ohrui, M., Kikuchi, H. and Terada, M.: Mining Association Rules Consisting of Download Servers from Distributed Honeypot Observation, *The 13th Int'l Conf. on Network-Based Information Systems* (*NBiS 2010*), pp.541–545 (2010).
[4] Rosyid, N.R., Ohrui, M., Kikuchi, H., Sooraksa, P. and Terada, M.: A Discovery of Sequential Attack Patterns of Malware in Botnets, *The 2010 IEEE Int'l Conf. on Systems, Man, and Cybernetics* (*SMC 2010*), pp.2564–2570 (2010).
[5] Hatada, M., Nakatsuru, Y., Terada, M. and Shinoda, Y.: Dataset for Anti-Malware Research and Research Achievements Shared at the Workshop, *IPSJ Malware Workshop 2009* (*MWS 2009*), pp.1–8 (2009) (in Japanese).
[6] Hatada, M., Nakatsuru, Y., Terada, M. and Shinoda, Y.: Datasets for Anti-Malware Research MWS 2010 Datasets, *IPSJ Malware Workshop 2010* (*MWS 2010*), pp.1–5 (2010) (in Japanese).
[7] Kuwabara, K., Kikuchi, H., Terada, M. and Fujiwara, M.: Heuristics for Detecting Botnet Coordinated Attacks, *The 4th Int'l Workshop on Advances in Information Security* (*WAIS 2010*), pp.603–607 (2010).
[8] Trend Micro Threat Encyclopedia: TSPY_KOLABC.CH Technical Details, available from ⟨http://about-threats.trendmicro.com/ArchiveGrayware.aspx?language=en&name=TSPY_KOLABC.CH⟩
[9] Rosyid, N.R., Ohrui, M., Kikuchi, H., Sooraksa, P. and Terada, M.: Analysis on the Sequential Behavior of Malware Attacks, *IEICE Trans. on Information and Systems*, Vol.E94-D, No.11, pp.2139–2149 (2011).
[10] Dagon, D., Zou, C. and Lee, W.: Modeling Botnet Propagation Using Time Zones, *Proc. 13th Network and Distributed System Security Symposium* (*NDSS*), pp.1–15 (2006).
[11] ThreatSTOP: ThreatSTOP + Juniper SRX Solves "A Big Headache for Us"!, posted on available from ⟨http://blog.threatstop.com/tag/use-case/⟩. (accessed 2012-01)
[12] Connors, M.: University of Baltimore, Threat STOP customer story, available from ⟨http://www.threatstop.com/customerstories⟩.
[13] Hu, Y. and Panda, B.: A data mining approach for database intrusion detection, *Proc. 2004 ACM Symposium on Applied Computing* (*SAC '04*), pp.711–716, ACM (2004).
[14] Ye, Y., Wang, D., Li, T. and Ye, D.: IMDS: Intelligent malware detection system, *Proc. 2007 Int. Conf. Knowledge Discovery and Data Mining* (*KDD'07*), pp.1043–1047 (2007).
[15] Wang, L., Tan, X., Pan, J. and Xi, H.: Application of PrefixSpan* Algorithm in Malware Detection Expert System, *1st International Workshop on Education Technology and Computer Science* (*ETCS'09*), Vol.3, pp.448–452 (2009).
[16] LaRosa, C., Xiong, L. and Mandelberg, K.: Frequent pattern mining for kernel trace data, *Proc. 2008 ACM Symposium on Applied Computing* (*SAC '08*), pp.880–885, ACM (2008).
[17] Burdick, D., Calimlim, M., Flannick, J. and Yiu, T.: MAFIA: A Maximal Frequent Itemset Algorithm, *IEEE Trans. Knowledge and Data Engineering*, Vol.17, No.11, pp.1490–1504, IEEE (2005).
[18] Brauckhoff, D., Dimitropoulos, X., Wagner, A. and Salamatian, K.: Anomaly extraction in backbone networks using association rules, *IEEE/ACM Trans. Netw.*, Vol.20, No.6, pp.1788–1799 (Dec. 2012).
[19] Takemori, K., Fujinaga, M., Sayama, T. and Nishigaki, M.: Host-based traceback; tracking bot and C&C server, *Proc. 3rd International Conference on Ubiquitous Information Management and Communication* (*ICUIMC '09*), pp.400–405, ACM (2009).

Masayuki Ohrui was born in Japan. He received his B.E. and M.E. degrees from Tokai University in 2009 and 2011. He has been working for Hitachi Ltd., Security & Smart ID Solutions Division since 2012. His interests include network security and intrusion detection.

Hiroaki Kikuchi was born in Japan. He received his B.E., M.E. and Ph.D. degrees from Meiji University in 1988, 1990 and 1994. After working in Fujitsu Laboratories Ltd. from 1990, in Tokai University from 1994, respectively, he joined Meiji University in 2013. He is currently a Professor at the Department of Frontier Media Science, School of Interdisciplinary Mathematical Sciences, Meiji University. He was a Visiting Researcher at the School of Computer Science, Carnegie Mellon University in 1997. His main research interests are fuzzy logic, cryptographic protocol, network security, and privacy-preserving data mining. He is a member of IEICE, the Japan Society for Fuzzy Theory and Systems (SOFT), IEEE and ACM. He is a fellow of IPSJ.

Nur Rohman Rosyid was born in Indonesia. He is received his Bachelor and Master degree, both in Electrical Engineering, from Gadjah Mada University, Yogyakarta, Indonesia, in 2002 and 2005, respectively. He received his Doctoral degree from Electrical Engineering, Faculty of Engineering, King Mongkut's Institute of Technology, Ladkrabang, Thailand (2012) sponsored by JICA/AUN SEED-Net. In 2002, he joined the University of Muhammadiyah Purwokerto as a lecturer until 2005. He is currently a lecturer at Electrical Engineering, School of Vocational, Gadjah Mada University. His research interests include chaotic systems in robotic, sensor network and network security.

Masato Terada was born in Japan. He received his M.E. in Information and Image Sciences from Chiba University, Japan, in 1986. He joined Hitachi, Ltd. in 1986. He is currently the Chief Researcher at Yokohama Research Laboratory, Hitachi. Since 2002, he has been studying at the Graduate School of Science and Technology, Keio University and received his Ph.D. in 2006. Since 2004, he has been with the Hitachi Incident Response Team. Also, he is a Visiting Researcher at the Security Center, Information - Technology Promotion Agency, Japan (ipa.go.jp), JVN associate staff at JPCERT/CC (jpcert.or.jp) and a Visiting Researcher at the Graduate School of Science and Engineering, Chuo University as well. He is a fellow of IPSJ.