PAPER Special Section on Information and Communication System Security

# Analysis on the Sequential Behavior of Malware Attacks*

**Nur Rohman ROSYID**[†a)], **Masayuki OHRUI**[††], ***Nonmembers***, **Hiroaki KIKUCHI**[††], **Pitikhate SOORAKSA**[†], ***and*** **Masato TERADA**[†††], ***Members***

**SUMMARY** Overcoming the highly organized and coordinated malware threats by botnets on the Internet is becoming increasingly difficult. A honeypot is a powerful tool for observing and catching malware and virulent activity in Internet traffic. Because botnets use systematic attack methods, the sequences of malware downloaded by honeypots have particular forms of coordinated pattern. This paper aims to discover new frequent sequential attack patterns in malware automatically. One problem is the difficulty in identifying particular patterns from full yearlong logs because the dataset is too large for individual investigations. This paper proposes the use of a data-mining algorithm to overcome this problem. We implement the *PrefixSpan* algorithm to analyze malware-attack logs and then show some experimental results. Analysis of these results indicates that botnet attacks can be characterized either by the download times or by the source addresses of the bots. Finally, we use entropy analysis to reveal how frequent sequential patterns are involved in coordinated attacks.

***key words:*** *PrefixSpan, malware, botnets, coordinated attack, sequential pattern*

## 1. Introduction

During the past decade, Internet malware threats have increased and have become extremely sophisticated; they are also becoming commercialized [1], [2]. Highly organized and coordinated attacks by botnets are able to make malicious activities such as distributed denial of service (DDoS), e-mail spam, and click fraud [2]. A set of infected and compromised computers (bots) connected to the Internet is controlled remotely by an unauthorized user (botmaster) and, if employed for nefarious purposes, is called a botnet [3]–[5]. To scale up the botnet, bots infect additional computers by sending malware, using various strategies such as self-replicating worms, email viruses, or password guessing. The botmaster can then send arbitrary commands to the botnet to take control of victims. These commands are issued using one of two control mechanisms [1], [4]. The first method involves a centralized architecture, in which the bots in the botnet communicate with a central Command-and-Control (C&C) server. Most botnets use Internet Relay Chat (IRC) servers as C&C servers. The C&C server forwards commands received from the botmaster to all bots in the botnet. In the second method, botnets use a distributed control mechanism via a Peer-to-Peer architecture.

Fortunately, botnet activity can be traced by the observation of malware footprints of several bots spread across the network. This method is used in the "honeypot" system. In [6], McCarty gave an example of the implementation of a honeypot that captures useful data about computer attacks in the network. The honeypot is a decoy host pretending to be a vulnerable computer that looks attractive to the attackers, i.e., a host dedicated to receiving attacks [7]. The honeypot records every inbound packet as an item in an access log, comprising Timestamps, Honeypot ID, Source/Destination port number, Source IP address, Source port number, Hash value (SHA1), Malware name**, and Malware file name.

In this work, we investigate 94 independent honeypots that have observed malware traffic on the Japanese tier-1 backbone. The observations were coordinated by the Cyber Clean Center (CCC). The CCC DATAsets for 2009 and 2010 comprise the access logs of attacks between May 1, 2008, and May 31, 2010. This paper explores and discovers coordinated attack patterns in the CCC DATAset. Because botnets employ systematic attack methods, the sequences of malware downloaded by the honeypots have particular forms of coordinated pattern.

Our contribution is to propose a new method for detecting the coordinated malware servers that are the source of frequent sequential attack patterns. We also give classification results and characteristics of the frequent sequential attack patterns. Our proposed method is based on a data-mining algorithm, namely the *PrefixSpan* algorithm of Pei et al. [8]. *PrefixSpan* is applied in this research because it is an algorithm for the efficient mining of sequential patterns in a huge dataset that avoids the requirement to construct candidate sets and makes low memory demands [9]. Analysis based on entropy is then conducted, following the widespread use of entropy analysis in network security fields such as DDoS attack detection [10], anomalously accessed IP packets [11], and identification of packet malware

Manuscript received February 1, 2011.
Manuscript revised June 3, 2011.

[†]The authors are with the Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, Chalongkrung Rd., Bangkok, 10520, Thailand.
[††]The authors are with the School of Science and Technology, Tokai University, Hiratsuka-shi, 259–1292 Japan.
[†††]The author is with the Hitachi Incident Response Team (HIRT), Hitachi, Ltd. Kawasaki-shi, 212–8567 Japan.

*An earlier version of this paper was published in IEEE SMC 2010, with the title "A Discovery of Sequential Attack Patterns of Malware in Botnets," by Nur Rohman Rosyid, Masayuki Ohrui, Hiroaki Kikuchi, Pitikhate Sooraksa, and Masato Terada.

a) E-mail: nrohmanr@ugm.ac.id; kikn@cs.dm.u-tokai.ac.jp

DOI: 10.1587/transinf.E94.D.2139

**The Malware name is derived from the malware signature used by commercial anti-virus software (Trend Micro).

Copyright © 2011 The Institute of Electronics, Information and Communication Engineers

executables [12]. In this paper, we claim that entropy is a measure of how evenly the sequential pattern is distributed in the honeypot system, with the highest entropy score corresponding to a pattern observed at most honeypots and the lowest score corresponding to a pattern observed at only a few honeypots.

The remainder of the paper is organized as follows. Section 2 introduces the concept of the *PrefixSpan* algorithm. Section 3 describes our framework for mining sequential attack patterns in malware and shows the relationships in the attack pattern, using source IP addresses and timestamps. The entropy-based analysis showing the classification of the sequential attack pattern from the viewpoint of commonly involved patterns of attack is presented in Sect. 4. We briefly discuss related works and potential applications in the Sect. 5. Finally, Sect. 6 concludes the paper.

## 2. The *PrefixSpan* Algorithm

Sequential pattern mining is a method for discovering subsequence patterns in a database. This method was introduced by Agrawal and Srikant [13] and described as follows. *Given a set of sequences, where each sequence comprises a list of elements and each element comprises a set of items, and given a user-specified minimum support threshold as a condition, sequential pattern mining aims to find all frequent subsequences, i.e., the subsequences whose occurrence frequency in the set of sequences is greater than or equal to the minimum support threshold*. A sequential pattern-mining method called Prefix-projected Sequential Pattern Mining (*PrefixSpan*), which discovers frequent subsequences as patterns in a sequence database, was initially proposed by Pei et al. [8].

Let $a_i, b_j$ be items and let $\alpha_i, \beta_j$ be sequences of items, where $\alpha = \langle a_1 a_2 \cdots a_n \rangle$ and $\beta = \langle b_1 b_2 \cdots b_m \rangle$. Then $\alpha$ is a **subsequence** of $\beta$, denoted by $\alpha \sqsubseteq \beta$, if and only if there exist integers $j_1, j_2, \cdots, j_n$ such that $1 \leq j_1 < j_2 < \cdots < j_n \leq m$ and $a_1 = b_{j_1}, a_2 = b_{j_2}, \cdots, a_n = b_{j_n}$. 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 $\alpha$ in a database $S$ is the number of tuples in the database containing $\alpha$, 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 $\alpha$ is called a **frequent sequential pattern** in database $S$ if the sequence is contained by 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, with a sequential pattern of length $\ell$ being called an **$\ell$-pattern**.

Let $\alpha$ and $\beta$ be sequences $\langle a_1 \cdots a_n \rangle$ and $\langle b_1 \cdots b_m \rangle$, respectively. In terms of the *PrefixSpan* algorithm, we identify the following.

1. **Prefix** and **Postfix**: sequence $\alpha$ is a prefix of $\beta$ if and only if $a_i = b_i$ for $i = 1, \cdots, m$. For example, $\langle a\ a\ b\ c \rangle$ is a prefix of $\langle \boldsymbol{a\ a\ b\ c}\ d\ d\ a\ b \rangle$. The sequence following a prefix is its postfix, $\langle d\ d\ a\ b \rangle$ in this case.

**Table 1** 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 |

**Table 2** Sequential patterns.

| Prefix | Projected Databases | Sequential Pattern |
|---|---|---|
| ⟨PE⟩ | ⟨WO TR⟩, ⟨TR WO⟩ | ⟨PE⟩:5 |
| | ⟨TR TS WO⟩, ⟨PE TR WO BK⟩, | ⟨PE TR⟩:5 |
| | ⟨WO TR WO⟩ | ⟨PE TR WO⟩:4 |
| | | ⟨PE WO⟩:5 |
| | | ⟨PE WO TR⟩:2 |
| ⟨WO⟩ | ⟨TR⟩, ⟨BK⟩ | ⟨WO⟩:5 |
| | | ⟨WO TR⟩:2 |
| ⟨TR⟩ | ⟨WO⟩, ⟨TS WO⟩, | ⟨TR⟩:5 |
| | ⟨WO BK⟩, ⟨WO⟩ | ⟨TR WO⟩:4 |
| ⟨BK⟩ | ⟨PE TR TS WO⟩ | ⟨BK⟩:2 |
| ⟨TS⟩ | ⟨WO⟩, ⟨PE PE TR WO BK⟩ | ⟨TS⟩:2 |
| | | ⟨TS WO⟩:2 |

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

As an example, consider the sequence database $S$ in Table 1. If the user specifies $min\_sup = 2$, then the sequential patterns in $S$ can be mined by the *PrefixSpan* method in the following 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 the support count.

**Step 2: Distribute the search space.**
The *projected database* can be distributed across the following five subsets according to the five prefixes that resulted from Step 1: (1) those having prefix ⟨PE⟩, ⋯, and (5) those having prefix ⟨TS⟩.

**Step 3: Find subsets of sequential patterns.**
These can be mined by constructing the five corresponding projected databases and repeating the process with each of them recursively.

Starting from prefix ⟨PE⟩, we can make a ⟨PE⟩-projected database that comprises five postfix sequences: ⟨WO TR⟩, ⟨TR WO⟩, ⟨TR TS WO⟩, ⟨PE TR WO BK⟩, and ⟨WO TR WO⟩. In a recursive procedure, we return to Step 1 by scanning the ⟨PE⟩-projected database once, finding all 2-pattern sequences having prefix ⟨PE⟩, namely ⟨PE WO⟩:5 and ⟨PE TR⟩:5. Then the ⟨PE⟩-projected database is divided into two subsets according to its two prefixes ⟨PE WO⟩ and ⟨PE TR⟩. Each generated projected database is then mined

recursively. With prefix ⟨PE WO⟩ having the three postfix sequences ⟨TR⟩, ⟨BK⟩, and ⟨TR WO⟩, mining these sequences results in the sequential pattern ⟨PE WO TR⟩, which cannot be scanned further because its support count is less than *min_sup*. With the prefix ⟨PE TR⟩ having the four postfix sequences ⟨WO⟩, ⟨TS WO⟩, ⟨WO BK⟩, and ⟨WO⟩, we have the resulting 3-pattern ⟨PE TR WO⟩:4. The final projected database and the sequential patterns are given in Table 2.

## 3. Sequential Patterns in Malware

### 3.1 Preprocessing Data

The CCC DATAset comprises the yearlong access logs of attacks generated by 94 independent honeypots. Each honeypot reboots every 20 minutes and, between each reboot, the honeypot records every inbound packet in an access log. The 20-minute period is called a time slot, or simply a "slot". In this work, we discover frequent attack patterns based on the sequences of malware downloaded by honeypots. To implement the *PrefixSpan* algorithm, we preprocess the data to form a text file. The text file comprises *lines*, with each line being a sequence of malware names. (The term *line* is interchangeable with *slot*, reflecting the terminology for honeypots used later in our discussion.) The timestamps for the downloaded malware determine the order of the malware within the slot. An example of preprocessed data is given in Table 3.

### 3.2 Sequential *n*-Patterns in Malware Attacks

#### 3.2.1 Coordinated Attacks

A botnet assault is associated with a highly organized, coordinated, and systematic strategy. Consequently, the sequence of malware downloaded by the honeypots must be in a particular form of coordinated pattern. To reveal the coordinated attack patterns, we use the number of malware items in a pattern, i.e., the length of the sequence/pattern. Figure 1 shows the distribution of the lengths of sequences in malware as the minimum support values are varied, where the *Y*-axis gives the number of patterns (in thousands). The distribution of the lengths of sequences tends to be between length 2 and length 3 on the *X*-axis. These fundamental features are useful in tuning the parameters in the data mining, i.e., they become a reference point for selecting the sequence length.

Mining of the CCC DATAset using the *PrefixSpan* algorithm produces a list of the sequential attack patterns of malware. The list is sorted according to the number of slots that are infected by malware. The accuracy of the minimum support determines the number of sequential attack patterns discovered. In this investigation, we select the minimum support for the sequential attack 2-patterns using the assumption that each honeypot reboots every 20 minutes, giving 72 slots per honeypot per day. If there are intensive attacks in a certain day, the number of slots infected will be less than or equal to 72. Let us suppose that 70 is reasonable as the minimum support for discovering sequential attack 2-patterns. Referring to the trend of the distribution of the lengths of sequences shown in Fig. 1, we choose 30 as the minimum support for 3-patterns, i.e., 40% of the minimum support for the sequential attack 2-patterns.

Now we show the overall sequential attack via the 3-patterns harvested from the CCC DATAset. Figure 2 shows the distribution of the average number of sequential patterns per day over two years. It indicates a trend in which the number of patterns increased significantly during 2010. This phenomenon should challenge those interested in network security to investigate further, particularly those involved in the analysis and automated detection of malware attacks.

#### 3.2.2 The Definition of Sequential Attack Patterns

The sequential attack patterns identified are indexed into the form $P_{x.y}$ to simplify the naming of patterns, where $x$ is the sequence length of the pattern and $y$ is its serial number in the naturally ordered list. As shown in Fig. 1, the minimum

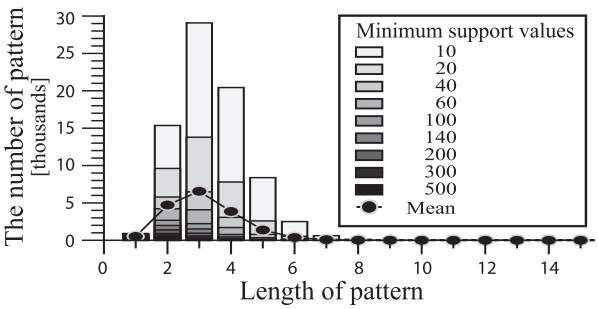



**Fig. 1** Distribution of the lengths of coordinated attack patterns.

**Table 3** Example of preprocessed data for a sequence database.

| Slot | Sequence of malware names |
|---|---|
| 0 | TROJ_SYSTEMHI.BQ |
| 1 | KDR_AGENT.ANHZ UNKNOWN TROJ_SYSTEMHI.BQ BKDR_AGENT.ANHZ UNKNOWN |
| 2 | PE_BOBAX.AH |
| 3 | PE_BOBAX.AH UNKNOWN BKDR_AGENT.ANHZ |
| ⋮ | |
| 15323 | PE_VIRUT.AV TROJ_IRCBRUTE.BW WORM_AUTORUN.CZU |
| 15324 | UNKNOWN PE_VIRUT.AV PE_VIRUT.AV WORM_AUTORUN.CZU TROJ_IRCBRUTE.BW |

support determines the number of patterns that result from pattern discovery. For the purposes of index naming, the minimum support needs to be reduced to 50% of that used for investigation purposes, namely 35 and 15 for 2-patterns and 3-patterns, respectively. As an example of index naming, pattern $P_{3.1203}$ is a sequential attack pattern with a sequence length of 3 and occupying line 1203 in the list.

Based on the form of the malware sequence, the mining results can be classified into two categories, *duplicate* and *nonduplicate*, where a duplicate pattern, unlike a nonduplicate pattern, has the same malware appearing more than once in it. For example, Fig. 3 (a) shows patterns $P_{2.386}$ and $P_{2.300}$, and Fig. 4 (a) shows patterns $P_{3.1203}$ and $P_{3.857}$. These patterns include the duplicated malware `PE_VIRUT.AV` and `PE_BOBAX.AK`.

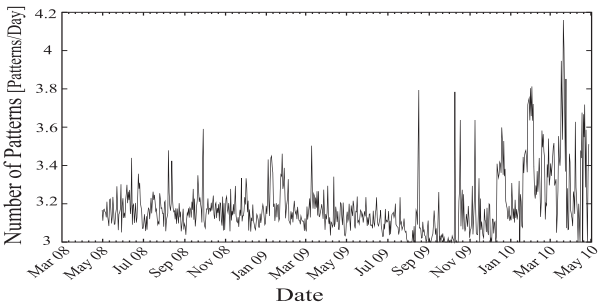



**Fig. 2** Distribution of the number of sequential attack patterns for 94 honeypots recorded by the CCC DATASet.

The behaviors of the sequential attack patterns for all honeypots seem to be similar. Therefore, we investigate just two of the 94 sample honeypots further, namely Honeypot 1 running under Windows XP+SP1 and Honeypot 2 running under Windows 2000. These two behaviors will be sufficient to represent the behavior of the sequential attack patterns in general. The results of mining the sequential attack 2-patterns for these two honeypots are depicted in the column-bars diagrams for Honeypots 1 and 2, respectively, shown in Figs. 3 (a) and 3 (b), where the $X$-axis is the pattern name and the $Y$-axis is the download frequency (in slots/year).

This study shows a significant relationship between these two honeypots in terms of the order in the list of the sequential attack patterns and the download frequency over a year. The five top-ranked sequential attack 2-patterns of malware from both honeypots have the same sequential pattern. As shown in Figs. 3 (a) and 3 (b), the other sequential attack 2-patterns have only small differences in the number of infected slots. The differences in the download frequency for each sequential attack pattern are relatively small. For example, the download frequencies of pattern $P_{2.453}$ for Honeypots 1 and 2 are 385 and 383 slots/year, respectively, which is a difference of just two slots/year.

The mining of the sequential attack 3-patterns for the two honeypots enables the extraction of 169 and 118 patterns (including 29% and 26% of nonduplicate patterns), re-

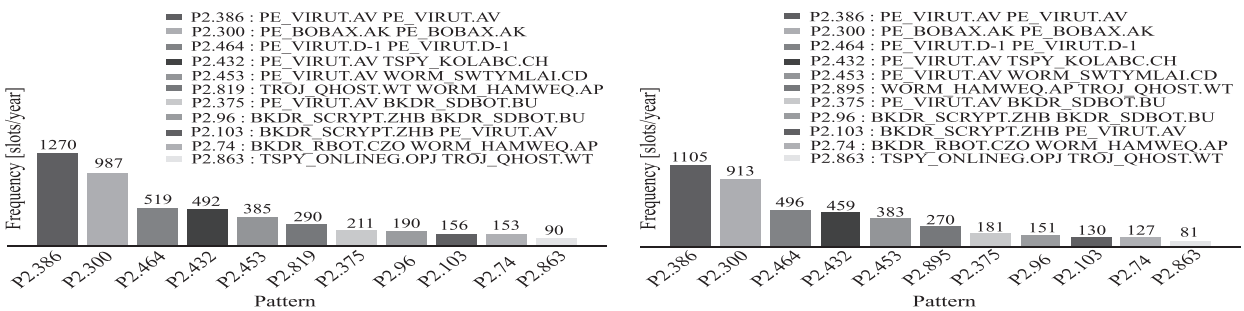



**Fig. 3** Sequential attack 2-patterns of malware.

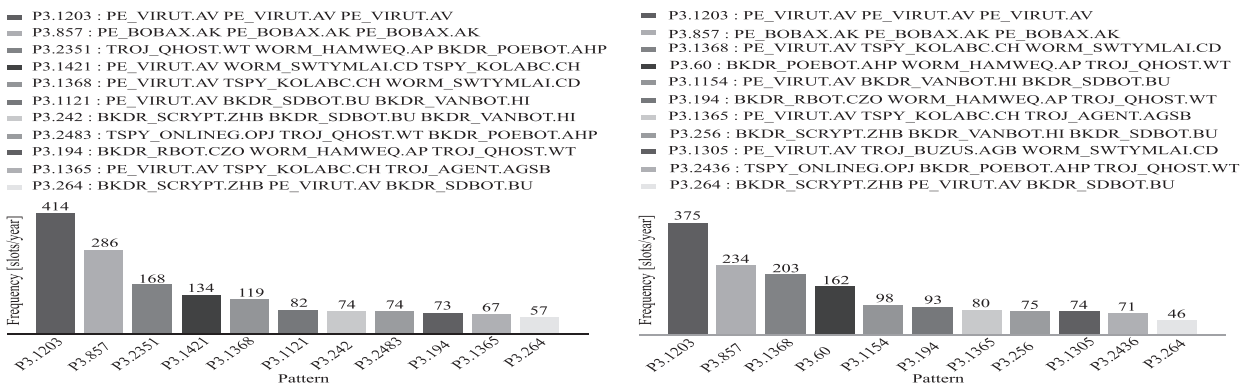



**Fig. 4** Sequential attack 3-patterns of malware.

spectively. Figure 4 shows the column-bars diagrams for the sequential attack 3-patterns for both honeypots. The top rankings of the lists for both sequential 2-patterns and 3-patterns are dominated by duplicate patterns that contain PE_VIRUT.AV and PE_BOBAX.AK. This duplication implies that the malware has successfully infected the honeypot more than once in a single slot. These facts may be regarded as an indication that these are the most common malware to have been employed by a botnet system.

### 3.2.3 Attack Pattern Based on IP Address and Timestamps

The botnet distributes malware to the bots via the Internet. Learning the behavior of the spread of malware from its source IP addresses and timestamps enables us to become alert to threats of botnet attack. For this purpose, we investigate the source IP addresses and malware timestamps that have been used by botnets.

First, we classify the sequential attack 3-patterns into several groups based on their source IP addresses and malware timestamps. Table 4 shows the naming of the sequential attack 3-patterns based on source IP addresses, where column *IP Pattern Code* lists the sequence of source IP addresses of malware. Table 5 shows the naming of the sequential attack 3-patterns based on malware timestamps, where column *Time Pattern Code* lists the timelines for the honeypots downloading the malware. For example, if pattern $P_{3.242}$ is of type $A_3E_3$, as shown in Table 6, the first and third malware items are downloaded from the same source IP address ($A_3$), and the second and third malware items are downloaded at the same time ($E_3$).

Source IP addresses can be used to distinguish the sources of malware. Some malware comes from a unique IP address and others from many IP addresses. The numbers of unique-host and IP-pattern types of the sequential attack 3-patterns are given in Table 6. Some of the sequential attack 3-patterns are of the single-source type, but others have two sources. Patterns in the top ranking are classified into two groups based on similarities in download times.

This paper identifies two groups that are worthy of investigation, *A* and *B*, as shown in Table 6. The attacker groups *A* and *B* differ in their source-IP pattern types. The sequential attack 3-patterns in Group *A* more often use the source-IP pattern types $A_1$, $A_4$ and $E_1$. Malware that compose patterns $P_{3.2351}$, $P_{3.194}$, and $P_{3.60}$ have downloaded simultaneously only from a unique host, even though the first malware item has downloaded from a few different unique hosts. As shown in the *unique host* column in Table 6, the leftmost field is the host of the first malware, followed by the second and third. One difference in the type of source-IP pattern in Group *A* is in the malware that composes each pattern. Note that TSPY_ONLINEG.OPJ is the first malware item that composes patterns $P_{3.2483}$ and $P_{3.2436}$, and it does not appear in the other patterns in Group *A*. This malware has downloaded from many different unique hosts, and at different download times from the second and third malware items. These patterns have 41 and 34 unique hosts at Honeypots 1 and 2, respectively.

The attacks made by the sequential attack 3-patterns in Group *B* of Table 6 often match the source-IP pattern types $A_3$, $A_5$, and $E_3$. All malware items in the sequential patterns were downloaded from different hosts. However, it seems only the first malware item has many unique hosts, whereas the second and third items have only a few unique hosts. This involves patterns $P_{3.1121}$, $P_{3.242}$, $P_{3.1154}$, and $P_{3.256}$. In Group *B*, pattern $P_{3.264}$ was downloaded by both honeypots, and the second malware item in this pattern has a different number of unique hosts. The first and third malware items in pattern $P_{3.264}$ have few unique hosts, but the second item PE_VIRUT.AV has many different unique hosts, as shown in the *unique host* column of Table 6. This can be considered as evidence that the botnet employs a collaboration and coordination strategy to attack victims.

**Table 4** Naming of attack patterns based on source IP address.

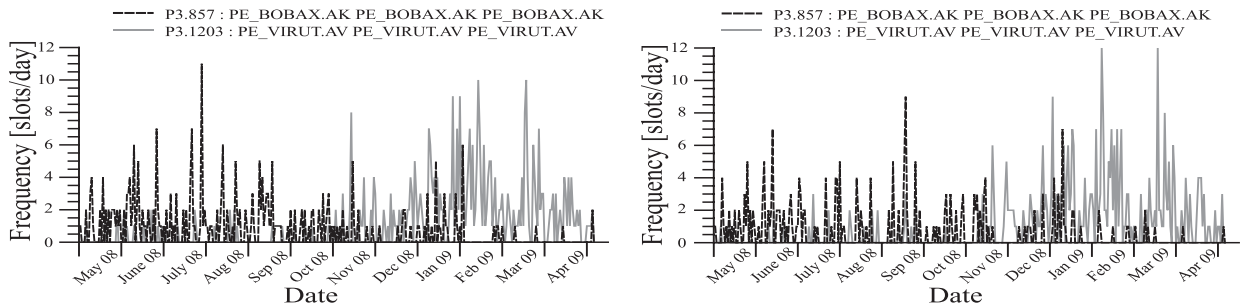
| IP Pattern Code | IP Pattern | | |
|---|---|---|---|
| $A_1$ | $S1$ | $S1$ | $S1$ |
| $A_2$ | $S1$ | $S1$ | $S2$ |
| $A_3$ | $S1$ | $S2$ | $S1$ |
| $A_4$ | $S1$ | $S2$ | $S2$ |
| $A_5$ | $S1$ | $S2$ | $S3$ |

**Table 5** Naming of attack patterns based on timestamps.

| Time Pattern Code | Time Pattern | | |
|---|---|---|---|
| $E_1$ | $T1$ | $T1$ | $T1$ |
| $E_2$ | $T1$ | $T1$ | $T2$ |
| $E_3$ | $T1$ | $T2$ | $T2$ |
| $E_4$ | $T1$ | $T2$ | $T3$ |

**Table 6** List of the sequential attack 3-patterns for the malware.

| #Honeypot | ID | Freq. | Sequential Attack Patterns | Ave[s] | SD[s] | Unique Host | | | Type | Gr. |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | $P_{3.2351}$ | 168 | TROJ_QHOST.WT WORM_HAMWEQ.AP BKDR_POEBOT.AHP | 4.27 | 51.07 | 1 | 1 | 1 | $A_1E_1$ | *A* |
| | $P_{3.2483}$ | 74 | TSPY_ONLINEG.OPJ TROJ_QHOST.WT BKDR_POEBOT.AHP | 97.04 | 165.46 | 41 | 1 | 1 | $A_4E_{1,3}$ | *A* |
| | $P_{3.194}$ | 73 | BKDR_RBOT.CZO WORM_HAMWEQ.AP TROJ_QHOST.WT | 56.65 | 235.71 | 3 | 1 | 1 | $A_1E_1$ | *A* |
| 2 | $P_{3.60}$ | 162 | BKDR_POEBOT.AHP WORM_HAMWEQ.AP TROJ_QHOST.WT | 34.12 | 175.92 | 8 | 1 | 1 | $A_1E_1$ | *A* |
| | $P_{3.2436}$ | 93 | TSPY_ONLINEG.OPJ BKDR_POEBOT.AHP TROJ_QHOST.WT | 72.66 | 191.33 | 34 | 1 | 1 | $A_4E_3$ | *A* |
| | $P_{3.194}$ | 71 | BKDR_RBOT.CZO WORM_HAMWEQ.AP TROJ_QHOST.WT | 381.48 | 478.60 | 5 | 1 | 1 | $A_1E_{1,3}$ | *A* |
| 1 | $P_{3.1121}$ | 82 | PE_VIRUT.AV BKDR_SDBOT.BU BKDR_VANBOT.HI | 108.31 | 212.90 | 48 | 1 | 1 | $A_3E_{1,3}$ | *B* |
| | $P_{3.242}$ | 74 | BKDR_SCRYPT.ZHB BKDR_SDBOT.BU BKDR_VANBOT.HI | 732.12 | 422.57 | 11 | 1 | 1 | $A_{3,5}E_3$ | *B* |
| | $P_{3.264}$ | 57 | BKDR_SCRYPT.ZHB PE_VIRUT.AV BKDR_SDBOT.BU | 862.60 | 304.87 | 5 | 42 | 1 | $A_5E_{3,4}$ | *B* |
| 2 | $P_{3.1154}$ | 98 | PE_VIRUT.AV BKDR_VANBOT.HI BKDR_SDBOT.BU | 75.54 | 177.64 | 55 | 1 | 1 | $A_5E_3$ | *B* |
| | $P_{3.256}$ | 75 | BKDR_SCRYPT.ZHB BKDR_VANBOT.HI BKDR_SDBOT.BU | 821.86 | 326.30 | 6 | 2 | 1 | $A_{2,5}E_3$ | *B* |
| | $P_{3.264}$ | 46 | BKDR_SCRYPT.ZHB PE_VIRUT.AV BKDR_SDBOT.BU | 968.42 | 258.12 | 6 | 34 | 1 | $A_5E_{3,4}$ | *B* |




(a) Honeypot 1 (b) Honeypot 2

**Fig. 6** Distribution of duplicate sequential attack 3-patterns of malware within a year.

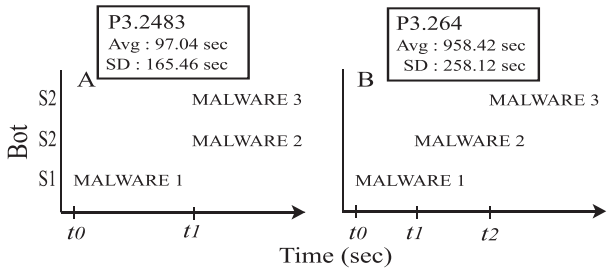



**Fig. 5** Time charts for coordinated attacks made by the sequential attack 3-patterns: (A) $P_{3.2483}$ belongs to IP pattern $A_4$ and time pattern $E_3$, (B) $P_{3.264}$ belongs to IP pattern $A_5$ and time pattern $E_4$.

Figure 5 shows time charts for coordinated attacks made by the sequential attack 3-patterns $P_{3.2483}$ and $P_{3.264}$. It illustrates how the coordinated attacks work. These are observed in terms of the source IP addresses and timestamps given in Tables 4 and 5. Consequently, naming and mapping the behaviors of sequential attack patterns may inform us about the spread of malware through the network, and lead us to identify and anticipate threats much earlier.

### 3.2.4 Distribution of Activity of Coordinated Attacks Over a Year

Figure 6 shows the distributions of the duplicate sequential attack 3-patterns that were most frequently downloaded by both honeypots, where the *X*-axis indicates the day of the year and the *Y*-axis indicates the download frequency in slots/day. The most common duplicate patterns are `PE_VIRUT.AV` and `PE_BOBAX.AK`. Sequential attack patterns are distributed uniformly during the year. As shown in Fig. 6 (a), pattern $P_{3.1203}$ has two peaks in February and March 2009 with 10 slots/day, whereas pattern $P_{3.857}$ is observed at the maximum rate of 11 slots/day at the end of July 2008. Similarly, Fig. 6 (b) shows that pattern $P_{3.1203}$ in Honeypot 2 peaks at a similar infection date as the same pattern in Honeypot 1, but with an infection rate of 12 slots/day. Pattern $P_{3.857}$ in Honeypot 2 has a peak in September 2008 of nine slots/day.

These two patterns in the honeypots are associated with malware that has the ability to disable some services on systems running Windows 2000 and Windows XP such as *Internet Connection Firewall* and *Internet Connection Sharing*. They listen to various ports and connect to an IRC server, and their potential for damage and propagation is rated as medium to high [14]. Regarding the botnet attack, we conjecture that the distribution diagram shown in Fig. 6 can be considered a distribution of the C&C activity of a botnet system.

### 3.2.5 Some Classes of Coordinated Attacks

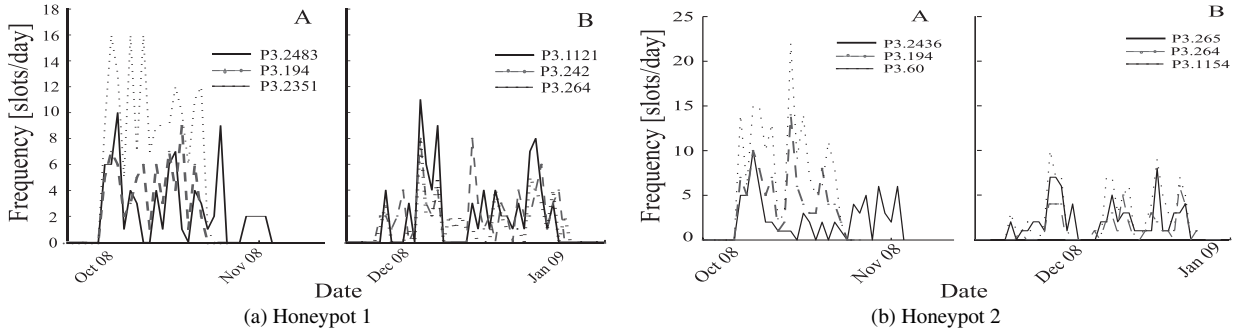
Nonduplicate sequential attack 3-patterns are similar either in the name of the malware or in the download time. Group *A* includes the same five malware names for both honeypots, but there are some differences in the sequence of malware patterns. Pattern $P_{3.194}$ appears in both honeypots, as shown in Table 6, but differs in download frequency by two slots/year.

Similarly, Group *B*, as shown in Table 6, includes four identical malware items for both honeypots, but their order is partially reversed. Pattern $P_{3.264}$ infects both honeypots but differs in download frequency by around 11 slots/year.

Figure 7 shows the distributions for nonduplicate sequential attack 3-patterns captured during one year and their classification into Groups *A* and *B*, as described above. Both honeypots observed Group *A*, and found that it was downloaded within a 20-day period in October 2008. The maximum infection rate for each honeypot is 16 slots/day and 22 slots/day, respectively, as shown in Figs. 7 (a-A) and 7 (b-A). The activity of botnet attacks in Group *B* looks like a sustained burst (see [17]) throughout 25 days. The activity ran from December 2008 to January 2009, and the maximum infection rate of 11 slots/day occurred at Honeypot 1, as indicated by Figs. 7 (a-B) and 7 (b-B).

This investigation found great similarities between the two honeypots. There are two common features of nonduplicate sequential attack 3-patterns. First, the pattern attacked intensively for a short period, less than a month in a whole year. Second, the number of slots infected is greater than the number of duplicate sequential attack 3-patterns.

In this work, we also investigate the time intervals for sequential attack 3-patterns downloaded by honeypots. The time interval is defined as the difference in time between the first and last malware item downloaded within a sequential attack 3-pattern. We show the average (*Ave*) and standard




(a) Honeypot 1 (b) Honeypot 2

**Fig. 7** Distribution of nonduplicate sequential attack 3-patterns of malware within a year.

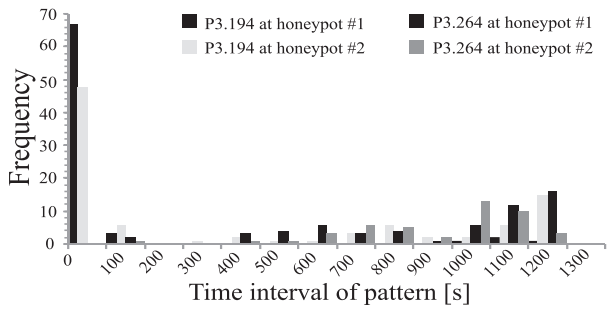



**Fig. 8** Histogram of time intervals for the sequential attack 3-patterns of malware.

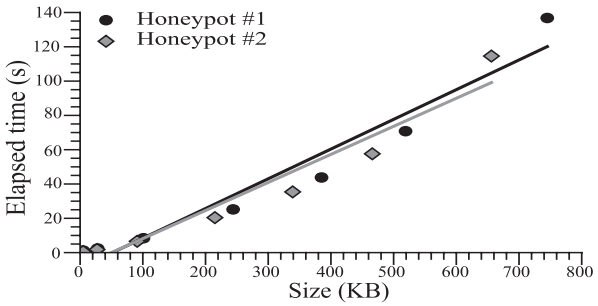



**Fig. 9** Performance of PrefixSpan method on mining sequential attack patterns.

deviation ($SD$) of time intervals for 3-patterns in Table 6. The distribution for each average time interval varies considerably. This may be a result of the dynamic behavior of Internet traffic causing gaps in the time interval for some download events or it may be caused by multiple botnet attacks.

Pattern $P_{3.194}$ in both honeypots has an average time interval of less than 7 minutes and a standard deviation greater than 7 minutes, but the distribution of its time interval, as shown in Fig. 8, indicates that the time interval mostly takes only a few values. This means that these patterns were executed at fixed constant time intervals, and it is therefore evidence that the patterns in Group $A$ were sent from the same botnet system.

Conversely, pattern $P_{3.264}$ in both honeypots has average time intervals greater than 14 minutes and standard deviations of less than 6 minutes, but the histogram of its time interval, in Fig. 8, indicates that the time interval is randomly spread and widely distributed. Therefore, we claim that these patterns in Group $B$ are an outcome of a collision of attacks made by a variety of botnets.

### 3.2.6 Performance of mining sequential attack patterns

We mined the sequential attack 2-patterns and 3-patterns using a machine with a 2.00 GHz *Intel*® *Core*™ 2 Duo T5750 CPU running the Ubuntu 10.10 Operating System with GCC version 4.4.5. The *PrefixSpan* algorithm was written in the C++ programming language.

Figure 9 shows the performance of *PrefixSpan* algorithm on mining sequential attack patterns. Performance test has involved pre-processing data with several size of files in two honeypots. The pre-processing data are varied from a day, a week, a month, 3 months, 6 months and a year which are represented in the $X$-axis. A general performance test has been taken as elapsed time of computation $s$ in the $Y$-axis. The slopes show the performance of PrefixSpan, 47.058 bps and 50.000 bps in honeypot #1 and #2, respectively.

## 4. Entropy Analysis of the Sequential Attack Patterns

Thousands of sequential attack patterns are generated by mining the CCC DATASet of 94 honeypots. The exposure of such a large quantity of valuable information from the discovery of sequential attack patterns raises another challenge. Classifying the sequential patterns in terms of entropy will help us to understand how common some patterns are in attacks on computer networks, and thereby identify significant behaviors in sequential attack patterns.

We first select some sequential patterns from among the thousands of pattern-mining results for the CCC DATASet. For each honeypot, the results of mining sequential patterns are sorted by download frequency. The selection is made based on the highest download frequency at each of the 94 honeypots for both duplicate and nonduplicate patterns.

The entropy of the sequential pattern $S$ in honeypots is defined by

$$H(S) = -\sum_{i=1}^{I} P(S_i)\log_2(P(S_i)), \quad (1)$$

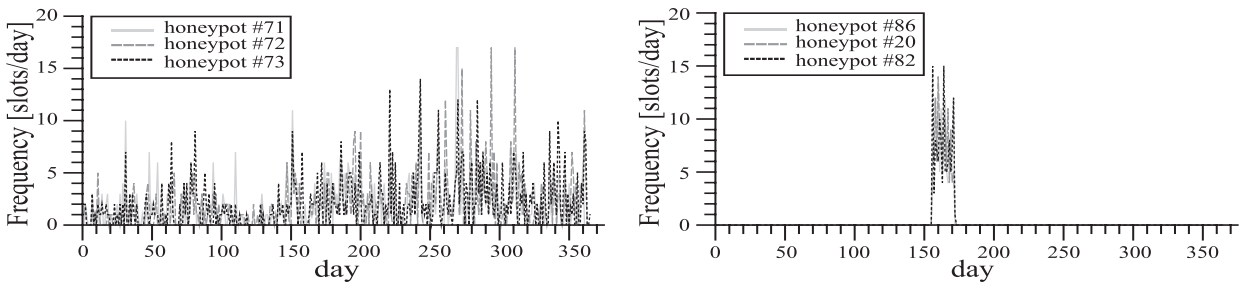
where $P(S_i)$ is the probability that the sequential pattern $S$ attempts to attack the $i$-th honeypot and $I$ is the number of honeypots. The probability that each sequential pattern attempts to attack a honeypot is the same. For example, if there are 10 honeypots infected by the sequential pattern $S$, then the probability of the sequential pattern is $P(S_1) = P(S_2) = P(S_3) = \ldots = P(S_{10}) = 0.1$. If the maximum number of honeypots $I$ is 94, then the entropy score will be in the range $0 \leq H(S) \leq \log_2(94)$. The results of the calculation are shown in Table 7, where the upper group of rows contains duplicate patterns, and the lower group of rows contains nonduplicate patterns.

The duplicate pattern $P_{3.1203}$ in the upper group and the nonduplicate pattern $P_{3.194}$ in the lower group of Table 7 have the highest entropy scores. This indicates that these patterns have attempted to attack the most honeypots. Further examination of their investigation attack distributions reveals that the two patterns have different characteristics. Figure 10 (a) shows that pattern $P_{3.1203}$ is distributed uniformly over one year. However, pattern $P_{3.194}$ has a narrow distribution at a specific date and a short duration, as shown in Fig. 10 (b). The behavior of pattern $P_{3.1203}$ suggests that malware involving this pattern is commonly used by several botnets. In contrast, pattern $P_{3.194}$ is seen at most honeypots, but it infects at a specific date and for a short period. Therefore, it can be deduced that pattern $P_{3.194}$ was sent by a particular botnet for a particular attacking purpose.

The duplicate pattern $P_{3.1924}$ in the upper group and the nonduplicate pattern $P_{3.2659}$ in the lower group each has a low entropy score, as shown in Table 7. The distribution of attacks for these kinds of patterns is seen for only one honeypot, as shown in Fig. 11. They are therefore probably either false detections of a pattern or accidental attacks by inexperienced users.

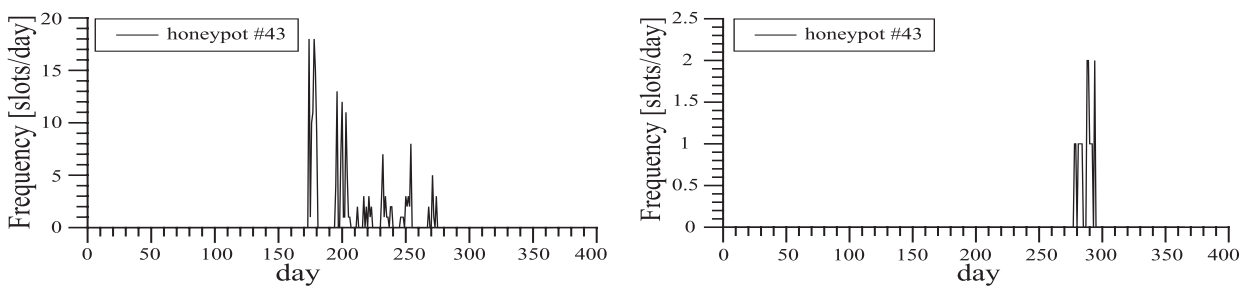
**Table 7** Entropy of the sequential attack patterns for all honeypots.

| ID | Pattern Name | Entropy |
|---|---|---|
| $P_{3.1203}$ | PE_VIRUT.AV PE_VIRUT.AV PE_VIRUT.AV | 6.0875 |
| $P_{3.2425}$ | TSPY_KOLABC.CH TSPY_KOLABC.CH TSPY_KOLABC.CH | 5.9307 |
| $P_{3.1590}$ | PE_VIRUT.D-4 PE_VIRUT.D-4 PE_VIRUT.D-4 | 5.8826 |
| $P_{3.857}$ | PE_BOBAX.AK PE_BOBAX.AK PE_BOBAX.AK | 5.8073 |
| $P_{3.1463}$ | PE_VIRUT.D-1 PE_VIRUT.D-1 PE_VIRUT.D-1 | 5.7814 |
| . | . | . |
| . | . | . |
| . | . | . |
| $P_{3.2796}$ | WORM_RBOT.GDJ WORM_RBOT.GDJ WORM_RBOT.GDJ | 2.0 |
| $P_{3.2528}$ | TSPY_ONLINEG.TKJ TSPY_ONLINEG.TKJ TSPY_ONLINEG.TKJ | 1.5850 |
| $P_{3.2676}$ | WORM_POEBOT.AKE TSPY_KOLABC.CH TSPY_KOLABC.CH | 1.0 |
| $P_{3.2611}$ | WORM_KOLABC.BQ PE_VIRUT.YE WORM_KOLABC.BQ | 0.0 |
| $P_{3.1924}$ | PE_VIRUT.YC PE_VIRUT.YC PE_VIRUT.YC | 0.0 |
| $P_{3.194}$ | BKDR_RBOT.CZO WORM_HAMWEQ.AP TROJ_QHOST.WT | 5.9307 |
| $P_{3.242}$ | BKDR_SCRYPT.ZHB BKDR_SDBOT.BU BKDR_VANBOT.HI | 5.7279 |
| $P_{3.2351}$ | TROJ_QHOST.WT WORM_HAMWEQ.AP BKDR_POEBOT.AHP | 5.6724 |
| $P_{3.134}$ | BKDR_POEBOT.GN TSPY_KOLABC.CH WORM_SWTYMLAI.CD | 5.5849 |
| $P_{3.1368}$ | PE_VIRUT.AV TSPY_KOLABC.CH WORM_SWTYMLAI.CD | 5.5546 |
| . | . | . |
| . | . | . |
| . | . | . |
| $P_{3.635}$ | BKDR_VANBOT.FM TROJ_PROXY.WE TROJ_PACK.DT | 1 |
| $P_{3.714}$ | BKDR_VANBOT.LE TROJ_BUZUS.ADZ WORM_SPYBOT.ADS | 1 |
| $P_{3.2336}$ | TROJ_QHOST.KY BKDR_RBOT.IA TROJ_VUNDO.MCS | 0 |
| $P_{3.2659}$ | WORM_POEBOT.AKE BKDR_POEBOT.GN TSPY_KOLABC.CH | 0 |
| $P_{3.2641}$ | WORM_PAKES.ABU PE_BOBAX.AK BKDR_VANBOT.LE | 0 |




(a) $P_{3.1203}$ (b) $P_{3.194}$

**Fig. 10** Distribution of sequential attack patterns that have a high entropy score within a year for three different honeypots: (a) pattern $P_{3.1203}$ and (b) pattern $P_{3.194}$.




(a) $P_{3.1924}$ (b) $P_{3.2659}$

**Fig. 11** Distribution of sequential attack patterns that have a low entropy score within a year for one honeypot: (a) pattern $P_{3.1924}$ and (b) pattern $P_{3.2659}$.

## 5. Discussion on Related Works and Its Potential Applications

### 5.1 Related Works

The study of datasets collected from honeypots provides valuable knowledge on the attacking behavior of botnets. These studies are classified into four categories; (1) heuristics technique, (2) N-gram, (3) clustering, and (4) PCA.

(1) Heuristic.

Heuristic techniques for the detection of malware involved in botnet coordinated attacks [15] provide useful information for determining the characteristics of and relationships between botnet attacks. However, heuristic approaches are ad hoc and therefore unable to adapt to any new attack.

(2) N-Gram.

In [16], Lu Wei et al., study on an automatic probing of botnet communities. Using cross-association clustering and *N-gram* algorithm for investigating the normal network traffic on a large-scale WiFi ISP network. Authors claim successfully achieve an automatic application for classifying network application communities and a generic method to distinguish malicious activity between human and bonets. Its process needs three steps to accomplish the goal. First, classify the input network flows based on payload signature to get the *unknown* flows. Next, examines the *unknown* flows based on cross-association clustering method to classify it into application communities. Finally, using *N-gram* algorithm to define whether the network flows are generated by humans or bots.

(3) Clustering traffic flows.

Thonnard and Dacier in [17], utilize graph-based clustering method for investigating datasets of Internet network traffic collected by honeypots based on similarities of time-signature. Authors have found that an appropriate similarity measure on time series analysis enables the identification of several worms and botnets activities. In other approaches, such as that of Gu et al. [18], a framework method is developed that uses clustering and cross-correlation techniques to identify a botnet. A filtering process is needed to eliminate unnecessary traffic flows. The next step is to cluster the flows based on similarities between malicious and communication activities. If a host belongs to both clusters, then it is strongly identified as part of a botnet. However, this method is not designed for the early detection of botnet attacks.

(4) Principal Component Analysis (PCA).

Because of their coordinated and systematic attacks, botnets generate activities with similar behavior patterns. In [19], Husna et al. investigate the behavior patterns of spammers based on the core similarities within spamming, particularly their temporal characteristics. Principal component analysis is applied to a feature set to determine which characteristics are important for the highest diversity in the spamming patterns, such as the active time, the content length, and the frequency of emails. Clustering methods are then used to classify spammers into botnet groups based on behavior similarities. The authors claim that this method enables us to recognize botnets precisely from their similar behavior when spamming a domain. However, a botnet is constructed by collecting as many Internet-connected computers as possible that are compromised by malware sent by the attacker. Whereas this method will struggle only against spamming generated by a botnet. This means that it may be less effective to anticipate the construction of the botnet in the network itself. In contrast, the classification of the sequential attack patterns of malware based on the time of sending by the attacker, as shown in Fig. 7, can identify the first day of botnet construction. By doing this, we can take action to anticipate the threat of the botnet at an early stage and thus derivatives threats from botnet can be eliminated.

The reason why these works are not able to work for sequential attacks is because of malware downloaded by honeypots have specific sequential patterns on various attributes such as malware name (based on hash value), source IP addresses and time interval among malware sequences. An important key point is *sequential pattern of malware* downloaded by honeypots in the continuous real time. Meanwhile all approaches [16]–[19] mainly utilize clustering methods for probing botnet attacks. Network flows are classified based on certain criteria such as application communities, source IPs, destination IPs, active time, content length, similar malicious activities and so on. Clustering process doesn't care enough to the sequences of malicious activity, but it more pay attention in the similarity of clustering criteria. Conversely, in fact, botnet attacks are established in a sequential manner. Then it becomes hard for detecting a new malicious acvitity or botnet attack use clustering method.

### 5.2 Some Potential Applications

Analysis results give us great challenges to explore the possible applications. Nowadays a lot of network security applications offer various methods to overcoming the dark side threats of the Internet. A lot of potential applications can be achieved from these results, and three of them will be explained such as a new Intrusion Detection and Prevention System (IDPS); a new botnets firewall which block out botnet sources; and tracking botnets which is possible to identify the sources of malware that sent by botnets and estimates the size of botnet.

Now we explain the corresponding potential applications above in the following description.

**IDPS** The result of the proposed method is considerable to

be a new type of the network-based IDPS. Because botnets can generate many kinds of attacks which hard for a specific IDPS technology for overcoming overall attacks. Thus, Monitoring and identifying suspicious activity based on the sequential attack patterns of malware downloaded by honeypot offers a wider scope of IDPS. It implies that if we can anticipate botnets, then we can eliminate many kinds of attacks which come from botnets.

**Botnet firewall** The classification and behaviors shown in Fig. 7 give valuable information as attack alerts. The attacker needs 20 to 25 days (as mentioned in Subsection 3.2.5) to establish an infected computer network as a botnet system. If the first day of infection by an attacker is identified, we can take action to prevent an ongoing threat. By mining periodically, we can obtain real-time statistics similar to those shown in Fig. 7. Therefore, the first day of infection can be identified by monitoring the download frequency, and it could be a start point to block out botnet sources.

**Tracking botnets** Due to botnets use systematic attack methods, the sequences of malware downloaded by honeypots have particular forms of coordinated pattern, as shown in Fig. 5. Those are valuable information which reveal specific attack for tracking botnets, where malware come from and how big the botnets are. Moreover, the sequential attack patterns of malware can explain botnet strategies how to compromise with victim machines. Source IP addresses belong to the sequential attack patterns of malware are considered as an evidence for network forensics.

## 6. Conclusion

Our analysis shows that coordinated attacks are performed by multiple sequential attack patterns within a short period. Malware used in a coordinated attack by sequential attack patterns has characteristics in the sequence with respect to either the download time or the source IP address of the servers. Entropy analysis helps us to discover the most common sequential attack patterns involved in coordinated attacks.

This paper reveals several behaviors that are useful for alerting users to botnet attack threats. We have found that the *PrefixSpan* method is sufficiently powerful for discovering and analyzing sequential attack patterns in honeypot systems.

## Acknowledgements

This research is supported by JICA AUN/SEED-Net under Collaborative Research (CR) Grant 2010-2011.

### References

[1] P. Wang, S. Sparks, and C. Zou, "An advanced hybrid peer-to-peer botnet," IEEE Trans. Dependable and Secure Computing, vol.7, no.2, pp.113–127, 2010.

[2] E. Hellweg, "When botnets attack," MIT Technology Review, Sept. 2004.

[3] M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach to understanding the botnet phenomenon," Proc. 6th ACM SIGCOMM Conference on Internet Measurement, IMC '06, pp.41–52, New York, NY, USA, 2006.

[4] N. Provos and T. Holz, "Tracking Botnets," Virtual honeypots: from botnet tracking to intrusion detection, pp.359–390, Addison Wesley Professional, 2007.

[5] H. Zeidanloo and A. Manaf, "Botnet command and control mechanisms," Computer and Electrical Engineering, ICCEE '09. Second International Conference, pp.564–568, 2009.

[6] B. McCarty, "Botnets: Big and bigger," IEEE Security & Privacy, vol.1, no.4, pp.87–90, 2003.

[7] L. Spitzner, Honeypots: Tracking Hackers, Addison Wesley, 2002.

[8] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.C. Hsu, "Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth," Proc. 17th Int. Data Engineering Conf, pp.215–224, 2001.

[9] F. Pedro, "A survey on sequence pattern mining algorithms." University of Informatics, Gualtar, Portugal., Jan. 2011, (available at http://alfa.di.uminho.pt/˜pedrogabriel/papers/SM_survey.pdf).

[10] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to DDoS attack detection and response," DARPA Information Survivability Conference and Exposition, 2003. Proc., vol.1, pp.303–314, 2003.

[11] T. Nakashima, S. Oshima, Y. Nishikido, and T. Sueyoshi, "Extraction of characteristics of anomalously accessed IP packets by the entropy-based analysis," Proc. Int. Conf. Complex, Intelligent and Software Intensive Systems CISIS 2008, pp.141–147, 2008.

[12] R. Lyda and J. Hamrock, "Using entropy analysis to find encrypted and packed malware," IEEE Security & Privacy, vol.5, no.2, pp.40–45, 2007.

[13] R. Agrawal and R. Srikant, "Mining sequential patterns," IEEE 11th International Conference on Data Engineering (ICDE'95), pp.3–14, 1995.

[14] Trendmicro, "Threat encyclopedia.," Nov. 2007 (available at http://about-threats.trendmicro.com/).

[15] K. Kuwabara, H. Kikuchi, M. Terada, and M. Fujiwara, "Heuristics for detecting botnet coordinated attacks," Proc. ARES '10 Int. Availability, Reliability, and Security Conf, pp.603–607, 2010.

[16] L. Wei, T. Mahbod, and G. Ali A., "Automatic discovery of botnet communities on large-scale communication networks," In ASIACCS '09: Proc. 4th Int. Symposium on Information, Computer, and Communications Security, pp.1–10, 2009.

[17] O. Thonnard and M. Dacier, "A framework for attack patterns' discovery in honeynet data," Digital Investigation, vol.5, no.Supplement 1, pp.S128–S139, 2008.

[18] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "BotMiner: clustering analysis of network traffic for protocol and structure-independent botnet detection," 17th Usenix Security Symposium (2008), 2008.

[19] H. Husna, S. Phithakkitnukoon, S. Palla, and R. Dantu, "Behavior analysis of spam botnets," Communication Systems Software and Middleware and Workshops, COMSWARE 2008. 3rd International Conference, pp.246–253, 2008.

**Nur Rohman Rosyid** was born in Indonesia. He received his B.Eng. and M.Eng. degrees, both in Electrical Engineering, from Gadjah Mada University, Yogyakarta, Indonesia, in 2002 and 2005, respectively. In 2002, he joined the University of Muhammadiyah Purwokerto as a Lecturer until 2005, when he moved to the Diploma Program of Electrical Engineering, Gadjah Mada University, as a junior Lecturer. His research interests include chaotic systems, complex networks, and network security. He is working toward his Ph.D degree, and is sponsored by JICA/AUN SEED-Net at the School of Computer Engineering, Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand.

**Masayuki Ohrui** was born in Japan. He is a graduate student in the Graduate School of Information Science and Engineering, Tokai University. 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, respectively. After working at Fujitsu Laboratories Ltd. from 1990 through 1993, he joined Tokai University in 1994. He is currently a Professor in the Department of Communication and Network Engineering, School of Information and Telecommunication Engineering, Tokai 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 protocols, and network security. He is a Member of the Japan Society for Fuzzy Theory and Systems (SOFT), the IEEE, and the ACM. He is a fellow of the Information Processing Society of Japan (IPSJ).

**Pitikhate Sooraksa** was born in Thailand. He is currently Associate Professor of Electrical Engineering at the School of Computer Engineering and Information Science, Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, Ladkrabang, Bangkok, Thailand. His research interests include IT-mechatronics, development of rapid prototypes in embedded systems, and computer-aided control. He received a B.Ed. (Hons) and M.Sc. in Physics from Srinakharinwirot University, an M.S. from George Washington University (1992), and a Ph.D. from the University of Houston (1996), both in Electrical Engineering.

**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 the Security Systems Research Dept., Systems Development Lab., Hitachi. From 2002, he studied at the Graduate School of Science and Technology, Keio University, receiving his PhD in 2006. Since 2004, he has been with the Hitachi Incident Response Team. In addition, 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 Research and Development Initiative Chuo University.