Vol.2010-DPS-142 No.37
Vol.2010-CSEC-48 No.37
2010/3/5

Regular Paper

Frequent Sequential Attack Patterns of Malware in Botnets

Nur Rohman Rosyid,[†1,†2] Masayuki Ohrui,[†1]
Hiroaki Kikuchi,[†1] Pitikhate Sooraksa[†2]
and Masato Terada [†3]

More than 90 independent honeypots have observed malware traffic at the Japanese tier-1 backbone. Typical attacks are made by multiple servers coordinating to send many kinds of malwares. This paper aims to discover some frequent new sequential patterns of malware attacks. It is not easy to identify particular patterns from a-year-long logs because the volume dataset is too large to investigate one by one. To overcome the problem, this paper proposes a data mining algorithm, *PrefixSpan* method. We implement the *PrefixSpan* algorithm to analyze the malware traffic and show the experimental result. The result of the analysis shows the sequential patterns of malware attacks tend to be change all the time.

1. Introduction

The malware is difficult to identify. The conventional attackers use integrated tools, called shellcode, containing; buffer overflow, port scan, trojan, worm, etc., to attacks their target. This technique is quite easy for antivirus software to anticipate the threats based on the hash signature, and the size of malware as well. Currently improved method splits the single malware into small parts of specific functions as malware and distributes them through the download server (DS) in the Internet. The DS is a host that has been already infected by malware. Afterward, the attacker uses the Command and Control (C&C) server to control the DSs to attack target. The attacker can manipulate and reconfigure their attacks according to their needs. The attackers usually utilize the Internet Relay Chat (IRC) server to send commands to DSs. This is how botnet system works. The attacks are coordinated systematically under the botnet attack strategy. In this paper, we call the sequential attacks by botnets *the coordinated attack*.

†1 Tokai University
†2 King Mongkut's Institute of Technology Ladkrabang
†3 Hitachi Incident Response Team (HIRT), Hitachi Ltd.

The conventional antivirus software based on the signature of single malware is not enough to detect the complicated and variety of the coordinated attack. One of the methods to identify a botnet's activity is observing the malware traffic distributed over several DSs on the network, using many honeypots. The honeypot is a decoy host pretending to be vulnerable computer and its looks attractive to the attackers. Honeypot will be rebooting every 20 minutes. During that time, every packet sent to honeypot is recorded as the access log consisting; Timestamp, Honeypot ID, Source/Destination port number, Source IP address, Source port number, Hash value(SHA1), Malware name, and Malware file name. The 20 minute duration is called a time slot, or simply slot.

More than 90 independent honeypots have observed malware traffic at the Japanese tier-1 backbone under coordination of the Cyber Clean Center (CCC). CCC DATAset 2009 consist of the access log of attack for a year during May 1, 2008 until April 30, 2009. In this paper, our interest is to explore and discover the coordinated attack pattern in the CCC DATAset 2009. Since botnet utilises systematic attack method, the sequence of malware downloaded by honeypots must be a particular form of coordinated pattern. This paper emphases to discover the frequent sequential attack pattern.

To achieve this goal, this paper applies a method based on a data mining algorithm, the *PrefixSpan* method1). Generally, this method is used to discover frequent sequential patterns in the transaction databases. Lina W. in[2)], used this method to discover the association rules of malware behavior pattern and combines with expert system. Ohrui3) use Apriori algorithm to find association rules of malware, which describes the confidence in the occurrence of a association rules of malware attacks. Methodology of Apriori is able to explore the sequential pattern but, it doesn't consider the timestamps. Moreover, *PrefixSpan* algorithm has the advantage from Apriori in term of memory consumption and computation cost4) and hence we choose *PrefixSpan* algorithm.

The rest of the paper is organized as follow. Section II introduces the basic concept of *PrefixSpan* algorithm. In Section III shows our framework for mining sequential attack pattern of malware. Section IV shows the relation of attack pattern and Source IP address and timestamp. Section V shows confidence of sequential attack pattern. Section VI concludes this paper.

2. Mining the Sequential Patterns

Sequential pattern mining is a method to discover subsequence patterns in database. This study was introduced by Agrawal R.[5] and this concept is described as follow: *Given a set of sequences, where each sequence consists of a list of elements and each element consists of a set of items, and given a user-specified minimum support threshold as a condition, sequential pattern mining is to find all of the frequent subsequences, i.e., the subsequences whose occurrence frequency in the set of sequences is greater than or equal to the minimum support*. Sequential pattern mining method, called *PrefixSpan* (i.e., Prefix-projected Sequential pattern mining) was firstly proposed by Jien Pei1), which discovers frequent subsequences as patterns in a sequence database.

Let a_i, b_j be items; α_i, β_j be sequences of item; $\alpha = \langle a_1 a_2 ... a_n \rangle$ 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 $j_1, j_2, ..., j_n$ such that $1 \leq j_1 < j_2 < ... < j_n \leq m$, 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 by at least min_sup tuples in the database, i.e., $support(\alpha) \geq min\_sup$. The number of item in a sequence is called the **length** of the sequence, so, sequential pattern with length ℓ is called **ℓ-pattern**.

Let us describe the *PrefixSpan* algorithm; Let α and β be sequences $\alpha = \langle a_1 ... a_n \rangle$ and $\langle b_1 ... b_m \rangle$.

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

(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, **c**-projection of $\langle a\ a\ b\ \mathbf{c}\ d\ c\ d\ a\ b \rangle$ is $\langle d\ c\ d\ a\ b \rangle$.

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		

(Example 1) Given a sequence database S in Table 1 and user specified $min\_sup = 2$, sequential patterns in S can be mined by *PrefixSpan* method in the following steps:

Step 1: Find 1-pattern sequence.
Scan database S once to discover all frequent items in sequence. These are ⟨`PE`⟩ :5, ⟨`WO`⟩:5, ⟨`TR`⟩:5, ⟨`BK`⟩:2 and ⟨`TS`⟩:2, where ⟨`pattern`⟩:*count* represents the pattern and its support count.

Step 2: Divide search space.
The database S can be partitioned into the following five subsets according to the five prefixes: (1) the ones having prefix ⟨`PE`⟩;...; and (5) the ones having prefix ⟨`TS`⟩.

Step 3: Find subsets of sequential patterns.
These can be mined by constructing corresponding projected databases recursively.

Starting from prefix ⟨`PE`⟩, let us generate ⟨`PE`⟩-projected database that consists of five postfix sequences: ⟨`WO TR`⟩, ⟨`TR WO`⟩, ⟨`TR TS WO`⟩, ⟨`PE TR WO BK`⟩, and ⟨`WO TR WO`⟩. Recursively, back to the step 1 by scanning ⟨`PE`⟩-projected database once, all 2-pattern sequences having prefix ⟨`PE`⟩ can be found, that is: ⟨`PE WO`⟩:5, ⟨`PE TR`⟩:5. Then ⟨`PE`⟩-projected database is divided into two subsets according to the two prefixes, *i.e.*, ⟨`PE WO`⟩ and ⟨`PE TR`⟩. Afterward, each generated projected database is mined recursively. From prefix ⟨`PE WO`⟩ having three postfix sequences ⟨`TR`⟩, ⟨`BK`⟩, and ⟨`TR WO`⟩, mining these sequences results sequential pattern ⟨`PE WO TR`⟩, which can not be scanned anymore because its frequency is too low. From prefix ⟨`PE TR`⟩ having four postfix sequences ⟨`WO`⟩, ⟨`TS WO`⟩, ⟨`WO BK`⟩, ⟨`WO`⟩, we have resulting 3-pattern ⟨`PE TR WO`⟩:4. The final projected database as

Table 2 Sequential pattern

Prefix	Projected Databases	Sequential Pattern
⟨PE⟩	⟨TR WO⟩, ⟨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

well as sequential patterns are listed in Table 2.

3. Mining Sequential Pattern of Malware

3.1 Input Data

We explore the CCC DATAset 2009 to discover frequent attack patterns based on the sequence of malware downloaded by honeypot. In this experiment, we investigate a year-long access log recorded by one of the honeypot out of 94 honeypots, which is honeypot **honey003**. For this purpose, we perform pre-processing access log so it is compatible to *PrefixSpan* algorithm. An input is a text file consisting lines of sequence of name of malware record in its timestamp of downloaded in one slot. The average of lines per honeypot is 15.324 lines (slots). The sample of the pre-processing data can be seen on Table 3.

3.2 List of Malware

We mine a list of malware from CCC DATAset 2009 with *min_sup* 1 and maximum length of pattern (*max_pat*) 1 to reveal the frequent sequence 1-pattern of malware. Running this experiment has a result of 537 malware variants classified into some categories; Trojan, Worm, Portable Executable (PE), Root Kit, Back door, etc. Figure 1 shows the top 10 list of malware that successfully has infected the honeypot.

As shown in Fig. 1, PE_VIRUT.AV and PE_BOBAX.AK are ranked at the top with

Table 3 Sample of pre-processing data of malware in a year (sequence database)

Slot	Sequence of Malware
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

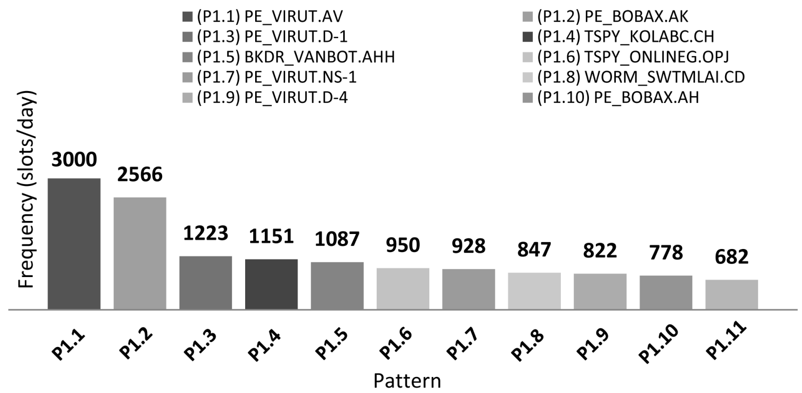


Fig. 1 Top 10 of malware

3000 (19.57%) and 2566 (16.75%) numbers of slot infected. Then followed by other kinds of malware with slot infected are less than a third of top two.

3.3 n-Pattern of Malware Attack

We discover the 2-pattern and the 3-pattern of malware coordinated attacks. To discover 2-pattern attack, we set *min_sup* 70 and *max_pat* 2. The average number of slots a day is 72 and hence we choose 70 as *min_sup*. Whereas *max_pat* 2 is a threshold maximum length of sequence.

Figure 2 shows a list of 2-pattern of malware. The sequence patterns are indexed of the form, $Px.y$, where x is a length of pattern and y is a serial number in the list. For example, $P_{2.1}$ is a 2-pattern of malware with serial number 1.

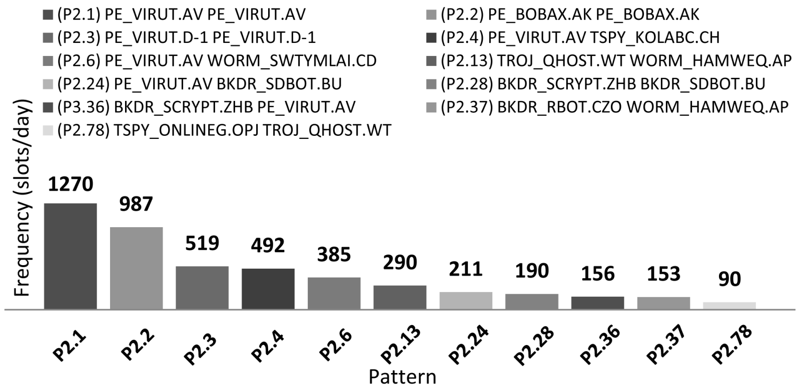


Fig. 2 List of 2-pattern attack of malware

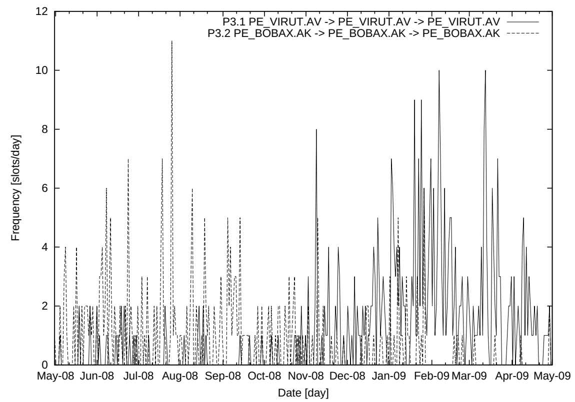


Fig. 3 Distribution of attacks of duplicate 3-pattern within a year

As shown in Fig. 2, there are two types of 2-pattern coordinated attacks, we call as *duplicate* and *non-duplicate* patterns. For example, patterns $P_{2.1}$ and $P_{2.2}$ observed as many as 1270 and 987 slots, respectively, are duplicate patterns of two malwares, `PE_VIRUT.AV` and `PE_BOBAX.AK`. It means that is at least more then one malware are duplicated in the pattern (n-pattern). Other patterns are called *non-duplicate* pattern. Top 3 in the list are dominated by the duplicate pattern. This duplication indicates that malware successfully infect honeypot more than one time in one slot.

Mining of 3-pattern with *min_sup* 30 (40% out of 2-pattern *min_sup*) extracts 169 3-pattern(s) including 29% non-duplicate patterns. Figure 3 shows the distribution of number of slot per day having 3-pattern ranked top 2. Both $P_{3.1}$ and $P_{3.2}$ are duplicate patterns of `PE_VIRUT.AV` and `PE_BOBAX.AK`. The average number of slots infected by $P_{3.1}$ and $P_{3.2}$ are 414 and 286, respectively.

As shown in Fig. 3, the patterns of these attacks are distributed uniformly for a year. Pattern $P_{3.1}$ has two peaks on Feb and Mar 2009 with 10 slots/day. Whereas pattern $P_{3.2}$ has been observed at the maximum rate of 11 slot/days on Aug 2008.

We investigate non-duplicate 3-pattern in top 50 list in Table 4. The six 3-patterns are divided into 2 groups of attack based on the time interval of their attacks; 3-pattern $P_{3.4}$, $P_{3.29}$ and $P_{3.30}$ on Oct through Nov 2008 as a group A and $P_{3.21}$, $P_{3.27}$, and $P_{3.49}$ on Nov 2008 through Jan 2009 as a group B. The three 3-patterns $P_{3.7}$, $P_{3.10}$, and $P_{3.37}$ are classify as group C and D. The distributions of attacks of groups A and B are shown in Fig. 4(a) and 4(b), whereas group C and D are shown in Fig. 4(c).

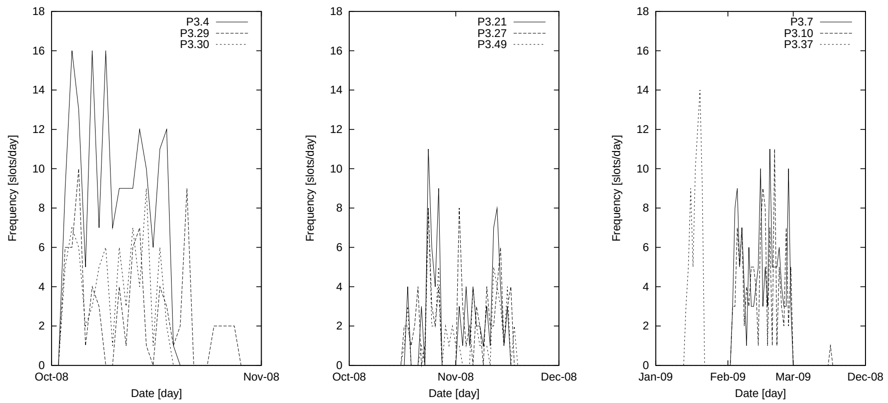
The attack patterns in group A are mostly assembled with five different kinds of malware, `TROJ_QHOST.WT`, `WORM_HAMWEQ.AP`, `BKDR_POEBOT.AHP`, `TSPY_ONLINEG.OPJ` and `BKDR_RBOT.CZO`. Maximum infection rate is 16 slots/day, which is three times of $P_{3.4}$ and the average is 6 slots/day. Group B has attack patterns assembled by four different kinds of malware, `PE_VIRUT.AV`, `BKDR_SDBOT.BU`, `BKDR_VANBOT.HI` and `BKDR_SKRYPT.ZHB`. The maximum infection rate is 11 slots/day carried out in pattern $P_{3.21}$ and the average is 3 slots/day. These groups of attack pattern are disjoint, *i.e.*, there is no malware used by both groups.

The group C are consisting 3-patterns $P_{3.7}$ and $P_{3.10}$ have two peaks of infection rate is 11 slots/day within 27 days of attacks on Feb through Mar 2009 and the average of infection rate is 4.7 slots/day. Pattern $P_{3.37}$ has attacks within very short of time on last of Feb 2009 around 8 days with 14 slots/day as a maximum infection.

The common features of non-duplicate 3-pattern attacks are; (1) these occurred

Table 4 List of the 3-pattern of botnets attack

Code	FREQ.	Sequential Attack Patterns			AVG Time	STD Dev	Unique Host			Pattern Type	Group
P3.1	423	PE_VIRUT.AV	PE_VIRUT.AVP	PE_VIRUT.AV	725.57	325.30	295	292	300	_ $E4$	–
P3.2	291	PE_BOBAX.AK	PE_BOBAX.AK	PE_BOBAX.AK	493.45	297.19	216	213	213	_ $E4$	–
P3.4	168	TROJ_QHOST.WT	WORM_HAMWEQ.AP	BKDR_POEBOT.AHP	4.27	51.07	1	1	1	$A1E1$	A
P3.29	74	TSPY_ONLINEG.OPJ	TROJ_QHOST.WT	BKDR_POEBOT.AHP	97.04	165.46	41	1	1	$A4E1$&$A4E3$	A
P3.30	73	BKDR_RBOT.CZO	WORM_HAMWEQ.AP	TROJ_QHOST.WT	56.65	235.71	3	1	1	$A1E1$	A
P3.21	82	PE_VIRUT.AV	BKDR_SDBOT.BU	BKDR_VANBOT.HI	108.31	212.90	48	1	1	$A3E1$&$A3E3$	B
P3.27	74	BKDR_SCRYPT.ZHB	BKDR_SDBOT.BU	BKDR_VANBOT.HI	732.12	422.57	11	1	1	$A3E3$&$A5E3$	B
P3.49	57	BKDR_SCRYPT.ZHB	PE_VIRUT.AV	BKDR_SDBOT.BU	862.60	304.87	5	42	1	$A5E3$&$A5E4$	B
P3.7	134	PE_VIRUT.AV	WORM_SWTYMLAI.CD	TSPY_KOLABC.CH	124.98	177.31	89	1	2	$A4E3$	C
P3.10	119	PE_VIRUT.AV	TSPY_KOLABC.CH	WORM_SWTYMLAI.CD	172.62	210.55	93	4	1	$A4E3$	C
P3.37	67	PE_VIRUT.AV	TSPY_KOLABC.CH	TROJ_AGENT.AGSB	163.43	200.34	45	42	4	$A5E3$	D



(a) Oct 2008 (b) Oct-Dec 2008 (c) Jan-Apr 2009

Fig. 4 Distribution of attacks non-duplicate of 3-pattern

intensively in the short time interval around one month a year, (2) the number of slots infected is greater than that of the attack with duplicate pattern.

We also investigate the distributions time interval of sequential 3-patterns to infiltrate into. Time interval is defined by a time difference between the first and last malware infections in the same sequential pattern at the honeypot. As shown in Table 4, we show the average time of interval of 3-pattern and its standard deviation. The distribution of average time varies well and hence we think that these attacks are caused by multiple botnets.

Figure 5 shows the distribution of time interval of the 3-pattern. As shown in Fig. 5, the variance of time interval of $P_{3.4}$ and $P_{2.9}$ tends to zero, which means these patterns are carried out in the fixed constant interval. This can be considered as an evidence that this 3-pattern of group A were sent by the same botnet system. In contrast, the time interval of $P_{3.27}$ and $P_{3.49}$ are widely distributed, therefore, we claim that 3-patterns of group B are the outcome of the collision of attacks by some botnets.

4. Attack Pattern Based on IP address and Timestamp

Botnet distributes malwares through the DS in the Internet. By learning the behavior of the spreading of malware through the source IP addresses and timestamps, we can highlight them as alerting of threats from botnets. For this purpose, we investigate the source IP address used by botnets and timestamp of

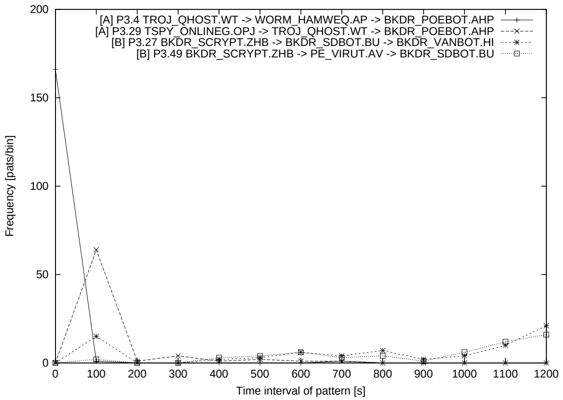


Fig. 5 Distribution of time interval of the 3-pattern

Table 5 Pattern attack based on source IP address

IP Pattern Code	IP Pattern		
$A1$	$S1$	$S1$	$S1$
$A2$	$S1$	$S1$	$S2$
$A3$	$S1$	$S2$	$S1$
$A4$	$S1$	$S2$	$S2$
$A5$	$S1$	$S2$	$S3$

Table 6 Pattern attack based on timestamp

Time Pattern Code	Time Pattern		
$E1$	$T1$	$T1$	$T1$
$E2$	$T1$	$T1$	$T2$
$E3$	$T1$	$T2$	$T1$
$E4$	$T1$	$T2$	$T2$
$E5$	$T1$	$T2$	$T3$

malware. First, we classify the patterns into several groups based on the source IP address and the timestamp. Table 5 and Table 6 show names mapping based on the source IP address and the timestamp, respectively. For example, $P_{3.27}$ has an IP pattern type of $A3E3$, *i.e.*, the first and third malware are downloaded from the same source ($A3$), the second and third malware are downloaded at the same time ($E3$).

We extract source IP addresses to distinguish distribution of malwares. Some malwares come from single unique source IP address and some from many source IP addresses. The unique host and pattern type of 3-pattern attack can be seen in Table 4.

As shown in Table 4, some 3-pattern has single IP pattern type, but some has two IP pattern types. Moreover, duplicate patterns are hard to be classified specifically, because malware classified in these patterns tend to spread from many host.

Groups of attacker, A and B as mention before, have different IP pattern types. The attacks by 3-pattern on group A often use IP pattern types $A1$, $A4$, and $E1$. Whereas, the attacks by 3-pattern on group B are classified in IP pattern type $A3$, $A5$, and $E3$. Groups C and D have similar timestamp pattern, that is $E3$.

5. Confidence of Sequential Attack Pattern

We want to know how reliable the sequential attack pattern is. The strength of sequential attack pattern is indicated by the degree of a confidence value, *i.g.*, how strength the n-pattern coordinated attack if $(n-1)$-pattern is subsequence of n-pattern occur. With the result of 1-pattern, 2-pattern, and 3-pattern generating by *PrefixSpan* algorithm, we evaluate the confidence of sequential attack pattern. To calculate the confidence, we use

$$for\ n > 1\ and\ m = (n-1), Conf(n\text{-}pattern) = \frac{Supp(n\text{-}pattern)}{Supp(m\text{-}pattern)}.$$

where n is the length of pattern and m is the length of subsequent of n-pattern.

For example, 3-pattern $P_{3.29}$ has subsequence 2-pattern $P_{2.78}$, such that, confidence of 3-pattern $P_{3.29}, i.e., Conf(P_{3.29}) = Supp(P_{3.29})/Supp(P_{2.78})$, so $conf(P_{3.29})$ is equal to 82.22%. It means 82.22% of sequential attack pattern of 2-pattern $P_{2.78}$ will be used to form a sequence 3-pattern $P_{3.29}$.

Figure 6 shows the confidence of 2-pattern attack. Confidence values of 2-pattern are always less than 50%. The highest confidence is reached by $P_{2.13}$ with 42.53% and the smallest one is $P_{2.78}$ with 9.4%.

Figure 7 shows the confidence of 3-pattern attacks. The highest and the smallest confidence are reached by $P_{3.4}$ and $P_{3.29}$ with 57.93% and 82.22%, respectively. The highest confidence that reached by $P_{3.29}$ is combined from the smallest of 2-pattern $P_{2.78}$ and the second highest of 3-pattern $P_{3.4}$ is combined from the highest 2-pattern $P_{2.13}$.

6. Conclusion

We have found that *PrefixSpan* method is sufficiently to discover all sequential

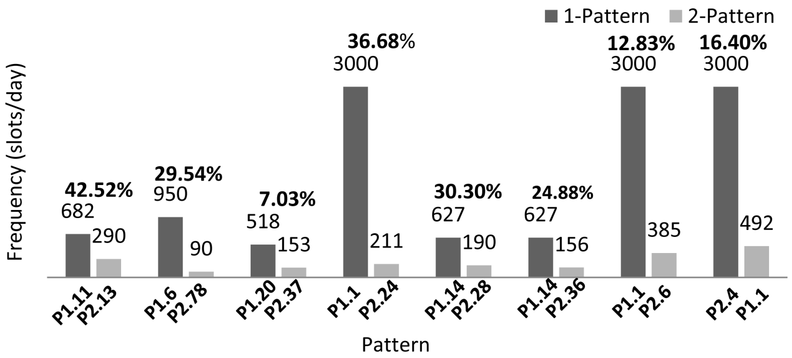


Fig. 6 Confidence (%) of sequential attack pattern 2-pattern

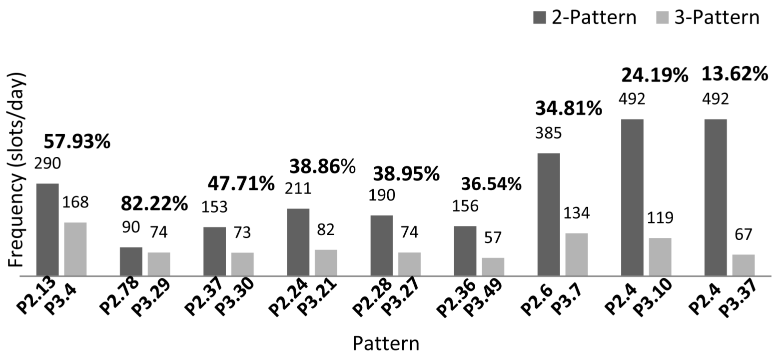


Fig. 7 Confidence (%) of sequential attack pattern 3-pattern

attack patterns. Our analysis shows that the coordinated attacks are performed by multiple sequential attack patterns within certain short time interval. The sequential pattern of coordinated attacks tends to change all the time. This paper gives several behaviors useful for alerting threats of botnets attacks.

References

1) J. Pei, et al., "PrefixSpan: Mining Sequential Patterns by Prefix-Projected Growth", in Proc. of The 17th Int'l Conf. on Data Engineering, pp.215-224, 2001.
2) W. Lina, et al., "Application of PrefixSpan* Algorithm in Malware Detection Expert System", in Education Technology and Computer Science (ETCS2009), pp. 448-452, 2009.
3) M. Ohrui, H. Kikuchi, and M. Terada, "Mining association rules consisting of download servers from distributed honeypot observation", in IPSJ Malware Workshop (MWS2009), pp.151-156, 2009.
4) F. Pedro "A survey on sequence pattern mining algorithms", in University of Informatics, Gualtar, Portugal, (http://alfa.di.uminho.pt/p̃edrogabriel/papers/SM_survey.pdf).
5) R. Agrawal and R. Srikant, "Mining Sequential Patterns", in Data Engineering, 1995. Proc. of The 11th Int'l Conf. on Data Engineering (ICDE95), pp.3-14, 1995.