IPSJ Journal Vol. 52 No. 9 2738–2747 (Sep. 2011)

Regular Paper

Principal Component Analysis of Botnet Takeover[⋆1]

Hiroaki Kikuchi,[†1] Shuji Matsuo[†1]
and Masato Terada[†2]

A botnet is a network of compromised computers infected with malware that is controlled remotely via public communications media. Many attempts at botnet detection have been made including heuristics analyses of traffic. In this study, we propose a new method for identifying independent botnets in the CCC Dataset 2009, the log of download servers observed by distributed honeypots, by applying the technique of Principal Component Analysis. Our main results include distinguishing four independent botnets when a year is divided into five phases.

1. Introduction

A botnet is a set of malicious software (malware) robots running in a distributed environment, under the control of the botnets' originator, called the "herder" or the "botmaster". The set of compromised hosts jointly perform attacks that look for vulnerabilities in a target network, and include "spamming", "phishing", "keylogging", click fraud, identity theft, and DDoS. These attempts are usually made via a specific destination port at which services with known vulnerable software are available. Ports 135, 138, and 445 are frequently scanned. There is also malware that uses particular ports to provide "back door" access to companies. Botnets often try to compete with each other to take over a network. According to Ref. 24), a Russian cybercrime organization uses "Spy Eye Toolkits", which remove their competitor's botnet software, known as "Zeus", from the victim's PC. Therefore, the compromised computers infected with malware may occasionally belong to different botnets.

According to a recent report in Ref. 21), multiple servers in a botnet collaborate to attack a single vulnerable host, aiming to take over compromised hosts that are under the control of another botnet master. Our study aims to clarify takeover activity involving several independent botnets, which can provide informative signatures for the botnets that can be used to trace them. Detecting botnet traffic, however, is not easy because botnets are evolving from a centralised strategy to a distributed strategy [23]. Many attempts have been made to analyze botnet traffic. Yegneswaran, et al. studied botnet control mechanisms in conjunction with host control commands [25]. Stayer, et al. proposed heuristics for detecting botnets based on flow characteristics such as bandwidth, duration of attacks, and packet timings [26]. Gu, et al. developed a system to automate the detecting process for botnet control channels. Their system, called "BotSniffer", used spatial–temporal correlation and similarity in network traffic.

In this paper, we study the CCC (Cyber Clean Center) Dataset 2009, which contains raw packet data captured from more than 90 independent "honeypots" over a two–year period. Our aim is to clarify the typical behavior of botnets from the observation of the CCC Dataset 2009.

The CCC, a Japanese governmental organization, observes the backbone of Japanese tier-1 providers using honeypots, which are virtual hosts running two guest operating systems that are periodically rebooted. The CCC Dataset 2009 provides 145 time slots in the period of time between reboots of the honeypot. These characteristics are useful for detecting and predicting *botnet coordinated attacks*. However, detecting these attacks remains incomplete because of the evolution of botnet strategy.

Our proposed scheme aims to identify the various botnets activities from distributed honeypot logs in an efficient manner. Our scheme is based on a *Principal Component Analysis* (*PCA*) of observations from logs. Under the simple assumption that a set of malware-infected hosts (bots) under the control of the same botmaster will work in coordination, our idea is to detect the dependencies between the coordinated activities and to decompose the accumulated vectors of multiple botnet activities into independent factors, which will identify the num-

†1 Tokai University
†2 Hitachi Ltd., Hitachi Incident Response Team (HIRT)

⋆1 The primary version of this work has been published in the 5th Joint Workshop on Information Security (JWIS 2010), Kikuchi, H. Matsuo, S. and Terada, M.: Principal Component Analysis of Botnets Takeover.

ber of major botnets. Our experiment with the CCC Dataset will answer these questions;

- How many independent botnets are used to take over a network?
- For how long does a single botnet try to attack its competitors?

The remainder of the paper is organized as follows. In Section 2, we present our proposed scheme after providing some fundamental definitions and models of botnets. In Section 3, we report on the experiment of using a real dataset involving botnet activity and show that the principal component basis of the dataset provides a significant characterization of botnet behavior. After presenting the experimental data, we discuss the expected number of independent botnets in Section 3.6.2. Finally, we make the concluding remarks in Section 4.

2. Proposed Method

2.1 Coordinated Attacks by Botnets

A typical botnet comprises the following hosts;

(1) A vulnerable web site,
(2) A "drive-by-download" server, from which a victim downloads an item of executable malware, and
(3) A command-and-control server that controls all infected hosts.

Under the control of the botmaster, a malicious individual, this group of infected hosts performs *coordinated attacks* that seek new vulnerable hosts to add to the botnet. This coordinated behavior makes it hard to trace back to the source of the attacks and to identify the botnet. According to Refs. 28), 29), coordinated attacks are observed as a sequence of downloading variety of malware shown in **Table 1**, where three malware are sent from different source addresses but with the same timing. This is the evidence that the downloads are performed in a programmed schedule in a botnet. This typical coordinated attack is written in the form

"$PE \to W \to T$",

where the order of downloads is indicated by arrows, with labels PE, W, and T, known as "Portable Executable", "Worm" and "Trojan horse", respectively.

A botnet typically uses multiple drive-by-download servers for each kind of malware, making the whole botnet more reliable, even though some of the hosts may be detected and blocked.

Table 1 Examples of coordinated infections [28].

slot	*time*	*srcIP*	*dstPort*	*MW*
0	0:02:11	124.86.A1.B1	47556	PE_VIRUT.AV
0	0:03:48	67.215.C1.D1	80	TROJ_BUZUS.AGB
0	0:03:48	72.10.E1.F1	80	WORM_SWTYMLAI.CD
2	0:36:46	124.86.A2.B2	33258	PE_VIRUT.AV
2	0:36:52	72.10.E1.F1	80	WORM_SWTYMLAI.CD
2	0:36:52	67.215.C1.D1	80	TROJ_BUZUS.AGB

2.2 Model of Competitive Botnets

Suppose two (competitive) botnets A and B have the sets of infected servers (S_1, S_2, S_3) and (S_4, S_5, S_6), respectively, dedicated to downloading the set of malware (PE, W, T). **Figure 1** illustrates a botnet coordinated attack and how often it downloads its malware. The servers perform scheduled download operations as shown in **Table 2**, which gives the average number of downloads per unit time. Under our hypothesis, the botnet constantly aims to find new vulnerable hosts to add to the controlled group, and even intercepts the existing servers belonging to another competing botnet. This is how a botnet tries to take over access to the Internet, and what we aim to clarify in this paper.

Under our hypothesis, botnet A was active in April while B was silent, both were very active in May and botnet B successfully obtained control of servers S_1, S_2, and S_3. Combining both of theses activity, results in the total number of downloads given by **Table 3**, where we can observe the effects of combining the two independent activities of A and B.

We can now define a problem:

(Problem) Given the accumulated numbers of observed downloads, estimate the degrees of activity for all botnets.

We can make a mathematical model of this problem. The above table can be written in terms of a corresponding 3-row, 6-column matrix X of the number of downloads;

$$X = \begin{pmatrix} 30 & 10 & 10 & 0 & 0 & 0 \\ 9 & 3 & 3 & 12 & 6 & 3 \\ 3 & 1 & 1 & 40 & 20 & 10 \end{pmatrix},$$

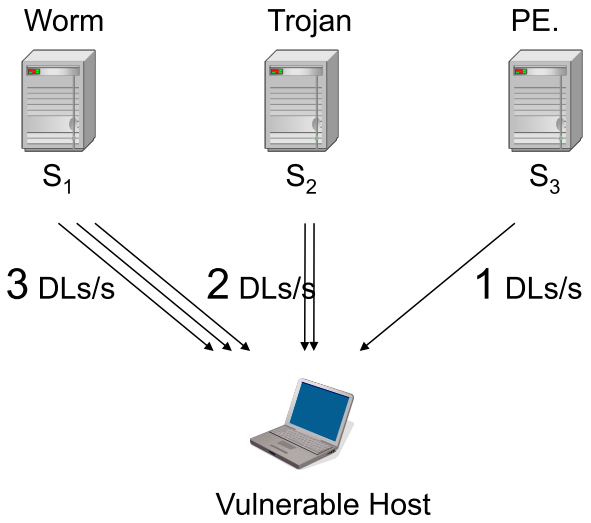

Fig. 1 A botnet coordinate attack (Host S_1 sends the Worm malware to the vulnerable host at a rate of three times a second. Host S_2 and S_3 make downloads twice and once per second on average, respectively. There are some reasons why Worm is assigned for server S_1. It is good for robustness, i.e., a single failure of server can be compensated by other servers. This makes it harder for us to trace attackers. It also makes it easier for the botnet to customize a set of malware.).

Table 2 Sample of the average number of downloads for two botnets.

	S_1	S_2	S_3	S_4	S_5	S_6
botnet A	3	1	1	0	0	0
botnet B	0	0	0	4	2	1

Table 3 Monthly statistics for downloads by six servers.

	S_1	S_2	S_3	S_4	S_5	S_6	
April	30	10	10	0	0	0	$10A$
May	9	3	3	12	6	3	$3A + 3B$
June	3	1	1	40	20	10	$A + 10B$

where each row vector is a linear combination of A and B, and corresponds to two independent botnets.

From the assumptions of linear algebra, an observation of sufficiently long duration determines the independent vectors from a sufficient number of redundant vectors.

We apply the following well-known technique to solve the problem. Let V be the covariance matrix of X, defined as $V = CC^\top$, where C is a normalized row vector of X by the average. The eigenvector of V efficiently gives the number of orthogonal and linearly independent vectors of C. In the case of the above table, the top two most significant eigenvalues have the following eigenvectors

$\boldsymbol{u}_1 = (-0.5, -0.2, -0.2, 0.7, 0.4, 0.2)$,

$\boldsymbol{u}_2 = (0.8, 0.3, 0.3, 0.5, 0.2, 0.1)$,

which recall the botnet activities A and B[⋆1]. The vectors, referred to as a *basis*, are orthogonal unit vectors, i.e., $\boldsymbol{u}_1 \cdot \boldsymbol{u}_2 = 0$ and $||\boldsymbol{u}_1|| = ||\boldsymbol{u}_2|| = 1$. From a well-known property of linear algebra, arbitrary vectors of X can be approximated by a linear combination of them. For example, the 1st-row vector $\boldsymbol{x}_1$ of X is written as:

$\boldsymbol{x}_1 = y_1\boldsymbol{u}_1 + y_2\boldsymbol{u}_2 + \boldsymbol{x}_0$

where $\boldsymbol{x}_0$ is a vector of mean values for each column of X, i.e., $\boldsymbol{x}_0 = (14, 4.6, 4.6, 17.3, 8.6, 4.3)$.

The property of orthogonality of the basis enables identification of the coefficients y_1 and y_2 simply as:

$y_1 = \boldsymbol{x}_1 \cdot \boldsymbol{u}_1 = -26$,

$y_2 = \boldsymbol{x}_2 \cdot \boldsymbol{u}_2 = 4$.

The coefficients y_1 and y_2 corresponding to the principal vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ imply the degree of power of the botnets. For example, the number of downloads in April is represented by a linear combination of two botnets with degrees of power -26 and 4, as;

$-26\boldsymbol{u}_1 + 4\boldsymbol{u}_2 \simeq (16.2, 6.4, 6.4, -16.2, -9.6, -4.8)$.

Consequently, we can identify a coefficient vector of the degrees of power for each botnet that approximates the given observations of downloads to an arbitrary level of accuracy. The competitive activity of botnets aiming to take over vulnerable servers results in changes to the coefficient vectors.

⋆1 Note the correspondence is $\boldsymbol{u}_1 = B - A$ and $\boldsymbol{u}_2 = A + B$.

3. Experimental Evaluation

3.1 Experimental Purpose

To show botnet activity in takeover attempts with respect to the degree of power for each botnet, we apply our proposed scheme to the CCC Dataset [20]. The experiment aims to clarify:

- The number of independent botnets,
- The duration of activities in the year.

3.2 Experimental Data

We made an analysis of the packet captured data in the CCC Dataset from several perspectives; (1) frequency of downloads per malware, (2) statistics of servers that send malware to honeypots, and (3) change of daily frequency of downloads. Our analysis shows 24 unique hash values from a total of 200 values, which are listed in **Table 4**. The unique hash values are identified with 13 kinds of unique malware names.

The fundamental statistics for the top 20 servers are given in **Table 5**, which lists the total quantity of malware downloaded, the active durations, the average number of downloads per day, the unique honeypots that observe the IP addresses, and the unique hash values, which is defined as:

Definition 3.1 A *hash value* is the 160-bit output of secure hash function, SHA1, to the entire binary image of malware. The number of distinct hash values in a given dataset is called *unique hash values*.

Table 4 List of malware and their statistics.

Malware ID	Label	Unique hash	DL counts	Scan counts (s4)	Protocol	Connection
PE_VIRUT.AV	PE1	8	91	18	TCP	PULL
PE_BOBAX.AK	PE2	1	4	4	TCP	PULL
PE_VIRUT.AT	PE3	1	1		TCP	PULL
BKDR_MYBOT.AH	BK1	1	1	6	UDP	PULL
BKDR_POEBOT.GN	BK2	1	30		TCP	PULL
BKDR_RBOT.ASA	BK3	4	5		UDP	PULL
TROJ_AGENT.ARWZ	TR1	1	6		TCP	PULL
TROJ_BUZUS.AGB	TR2	1	24		TCP	PULL
WORM_ALLAPLE.IK	WO1	1	1		TCP	PUSH
WORM_POEBOT.AX	WO2	1	1		TCP	PULL
WORM_SWTYMLAI.CD	WO3	1	27		TCP	PULL
WORM_AUTORUN.CZU	WO4	1	3		TCP	PULL
WORM_IRCBOT.CHZ	WO5	1	1		TCP	PULL
UNKNOWN	UK	1	5		TCP	PULL

Figure 2 demonstrates the frequencies of downloads for the top four source addresses in a year. Note that there is no server that has been operating for the whole year. Most servers operate for a few months.

3.3 PCA

There are four processing steps as follows:

(1) Given a matrix X of $m = 365$ rows and $n = 100$ columns [⋆1], compute $\boldsymbol{x}_0$, a matrix of mean values, defined by $(x_{00}, \ldots, x_{0n})$, where $x_{0j} = 1/m \sum_i^m x_{ij}$.

(2) Compute the covariance matrix of X, defined by $V = C \cdot C^\top$, where C is the normalized row vector $C = X - \boldsymbol{x}_0$. Compute the eigenvalue of V, $\lambda_1, \ldots, \lambda_n$, and the eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n$, sorted by order of eigen-

Table 5 List of drive-by-download servers sorted by frequency of downloads.

Rank	IP address	Total [DLs]	Duration [days]	Average [DLs/day]	Unique hash	Unique honeypots
1	AAA.10.167.74	462246	184	2512.21	119	91
2	AAA.10.166.195	399562	249	1604.67	48	92
3	BBB.114.143.2	33283	73	455.93	29	82
4	BBB.114.141.207	32202	53	607.58	37	78
5	CCC.215.1.206	26780	62	431.94	7	59
6	DDD.95.79.6	19641	117	167.87	99	85
7	AAA.10.169.26	14951	223	67.04	52	82
8	EEE.48.75.63	11699	148	79.05	100	69
9	CCC.18.161.250	10060	121	83.14	131	68
10	AAA.8.143.164	5099	70	72.84	40	81
11	FFF.202.252.41	4901	43	113.98	13	87
12	GGG.131.76.60	4659	71	65.62	52	74
13	CCC.215.1.226	4492	76	59.11	36	68
14	HHH.247.2.38	4262	61	69.87	166	77
15	III.18.116.75	4112	128	32.13	6	75
16	JJJ.219.170.67	3963	106	37.39	56	72
17	KKK.16.245.53	3766	31	121.48	9	63
18	LLL.90.134.24	3723	66	56.41	15	83
19	MMM.180.151.74	3473	29	119.76	6	87
20	HHH.247.2.32	3368	29	116.14	11	82

⋆1 The reason why we choose $n = 100$ is the power low distribution of the number of downloads per server, or *long-tail*. As Table 5 shows, a small number of top ranked servers occupy the entire downloads, e.g., the top two servers involve 75% of downloads, top-10 has 88%. Hence, we decide $n = 100$ is sufficient to cover the entire downloads.

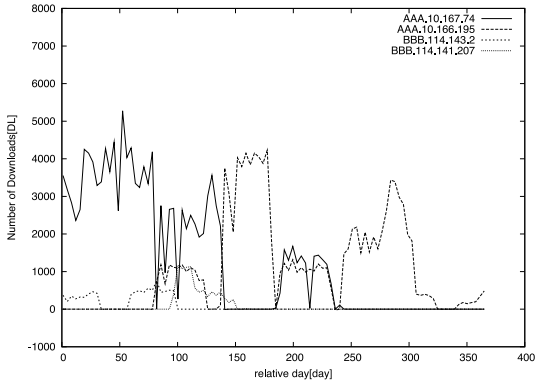

Fig. 2 Daily frequency of downloads over time, for the top four servers, identified by their source IP addresses.

value.

(3) Perform the orthogonal expansion of X using the eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$ as orthogonal bases, i.e., for $i = 1, \ldots, 100$, compute the principal components $y_1, y_2, \ldots, y_n$ as

$$y_i = \boldsymbol{x}_i \cdot \boldsymbol{u}_i.$$

(4) Plot a 365-day observation matrix on the reduced space of (y_1, y_2), where any particular change can be seen as the outcome of a botnet takeover. Classify the year into several phases in terms of the most significant botnet.

3.4 Experimental Results

Table 6 shows the four orthogonal bases, the eigenvectors of the servers, resulting from the PCA analysis of matrix X. **Figure 3** shows the top 20 eigenvalues of convergence matrix. The principal components, y_1 and y_2, over the year are shown in **Fig. 4**, where the year is divided into five phases. **Figure 5** is a scatter plot of 365 observations, with the horizontal axis y_1 and the vertical axis y_2. Similarly, **Figs. 6**, **7** and **8** show the distribution of principal components, y_3 and y_4, the distribution of y_2 and y_4, and the scatter plot of y_2 and y_4, respectively.

3.5 Processing Time for Analysis

We perform our analysis using GNU Octave 3.0.3, a high-level interpreted language designed for the numerical solution of linear and nonlinear problems and for numerical experiments. **Figure 9** shows the processing time for performing the entire steps described in Section 3.3, performed in Intel(R) Core(TM) Duo CPU T2400, 1.83 GHz, 2.00 GB memory, running Windows Vista(TM) Business, SP2. The figure illustrates how much the scheme scales with respect to the

Table 6 Principal component bases (snipped).

IP address	Principal components			
	$\boldsymbol{u}_1$	$\boldsymbol{u}_2$	$\boldsymbol{u}_3$	$\boldsymbol{u}_4$
A.10.167.74	**−0.83**	0.54	−0.02	0.08
A.10.166.195	0.55	**0.83**	−0.02	0.02
B.114.143.2	−0.05	0.02	−0.31	**−0.90**
B.114.141.207	−0.02	0.03	**0.94**	−0.27
C.215.1.206	0.01	−0.11	−0.03	0.28
D.95.79.6	0.02	−0.02	−0.04	0.11
A.10.169.26	−0.01	0.02	−0.04	−0.01
E.48.75.63	0.01	0.01	−0.03	0.04
C.18.161.250	0.01	−0.01	−0.02	0.05
A.8.143.164	−0.01	0.00	0.01	−0.09
F.202.252.41	0.01	0.00	−0.01	0.00
G.131.76.60	−0.01	0.01	−0.03	0.00
C.215.1.226	0.00	−0.01	−0.01	0.03
H.247.2.38	0.00	−0.01	−0.01	0.03
I.18.116.75	−0.01	0.01	0.00	−0.03
J.219.170.67	−0.01	0.00	0.04	−0.04
K.16.245.53	0.00	0.01	0.03	0.01
L.90.134.24	0.00	0.00	0.00	0.01
M.180.151.74	0.00	0.00	−0.01	0.01

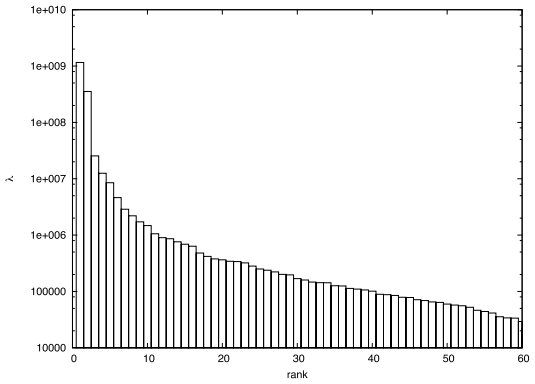

Fig. 3 Distribution of the top 20 eigenvalues λ.

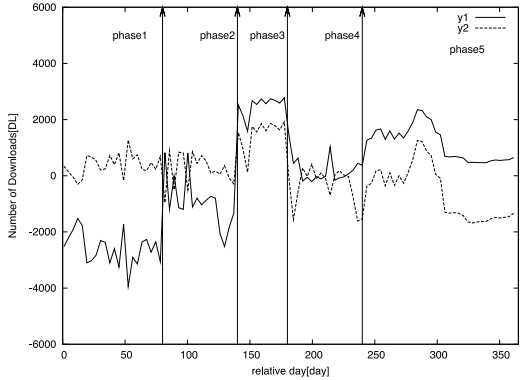

Fig. 4 The 1st and 2nd principal components over time, classified into five phases.

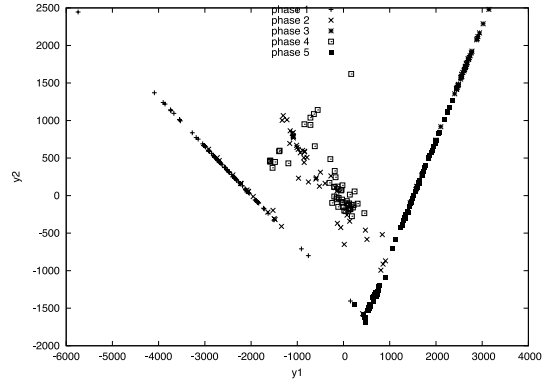

Fig. 5 Scatter plot of the 1st and 2nd principal components.

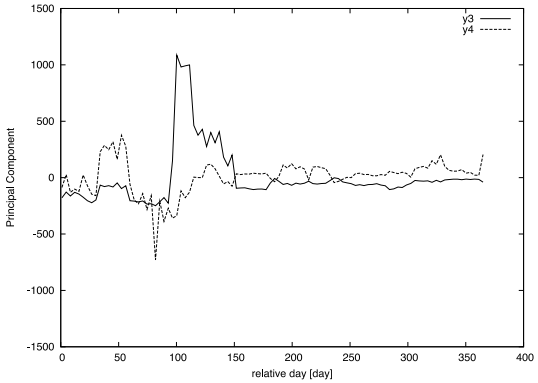

Fig. 6 The 3rd and 4th principal components over time.

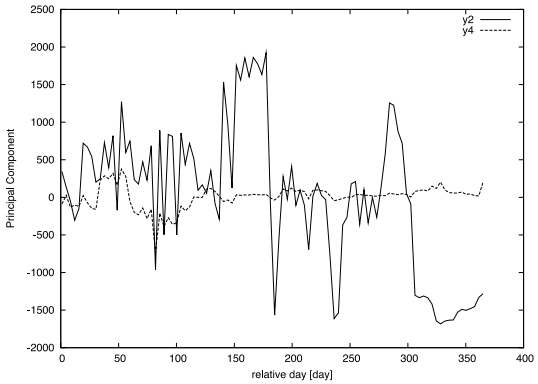

Fig. 7 The 2nd and 4th principal Components over time.

number of downloading servers.

3.6 Remarks

3.6.1 Phase Classification

We classified one-year of observation days into five phases in Fig. 4. A phase is a duration when servers perform downloads in similar ways within the duration and the phase is terminated when a totally different behavior of downloads are observed. The classification into phases can be made based on the relationship among principle components. For instance, we show the dominant conditions for a phase to be classified in **Table 7**, where the linear relation between the first and the second principle component, y_1 and y_2, are given via fitting. R-squared value R^2 shows the correlation coefficients for every phase. The R^2 for phases 1, 3 and 5 are very high, i.e., the servers perform downloads in a certain ratio unchanged within the phase. For easier understanding, we illustrate the conditions of y_1 and y_2 for phases with lines in the scatter plot in **Fig. 10**.

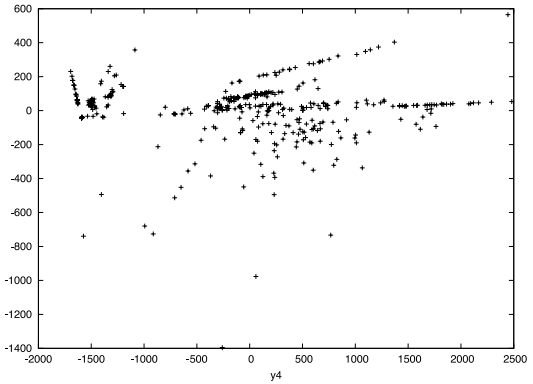

Fig. 8 Scatter plot of the 2nd and 4th principal components.

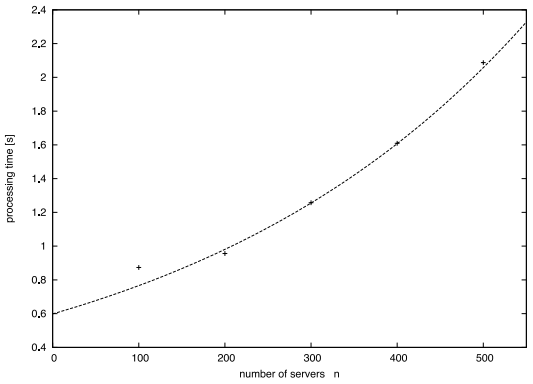

Fig. 9 Processing time for analysis.

Note that not all days are sharply classified into phases. In the figure, phases 2 and 4 have uncertain boundaries, and lower correlations in Table 7. However, phase 2 is an intermediate and transitional state between two clear phases 1 and 3. Phase 4 is between phases 3 and 4, as well. Hence, the process of phase classification is deterministic.

Throughout the work, we have made the phase classification manually, but we are positive that an automated classification is feasible because of the clear

Table 7 List of main infection phases.

Phase	Duration	Equation	R^2
1	2008/5/1 – 2008/7/19	$y_2 = -0.65y_1 - 1300$	1.0
2	2008/7/20 – 2008/9/17	$y_2 = -0.015y_1 + 225$	0.0007
3	2008/9/18 – 2008/10/27	$y_2 = 1.5y_1 - 2275$	0.999
4	2008/10/28 – 2008/12/26	$y_2 = -0.90y_1 - 209$	0.322
5	2008/12/27 – 2009/4/30	$y_2 = 1.5y_1 - 2275$	0.999

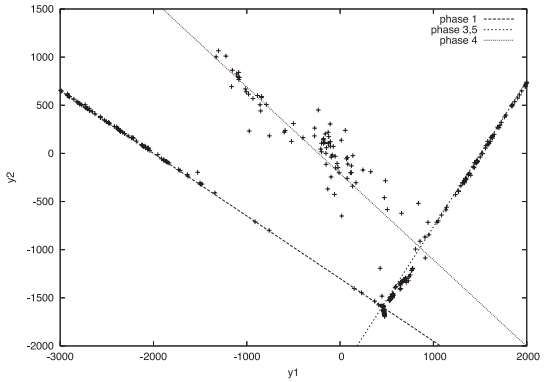

Fig. 10 Phase and dominant relationship between principle factors.

conditions of the phase.

3.6.2 Number of Independent Botnets

Figure 5 shows a linear relationship plot for phases 1, 3, and 5, which implies that there are two major botnets competing to take over the remaining servers. The observation gives us insight into a way to identify the independent activity of botnets via lines in a scatter plot. Our hypothesis is consistent with other scatter plots in Fig. 8. Above all, we can conclude that there are four independent botnets, agreeing with our first analysis.

Unfortunately, the confidence in our analysis is not high because the 20 eigenvalues are distributed with only slight decreases, as shown in Fig. 3, and sharp boundaries are therefore hard to identify. There may also be influence from a skewed distribution of honeypots.

Figure 5 shows the 5 phases of botnet activity. The average duration is 2 months. In other words, every two month a botnet takes over the network where

our honeypots are distributed.

3.7 Takeover Activities

From the observations, we can see the trade-off of activities between y_2 and y_1. We consider that this behavior was caused by the two competing botnets taking over the major server. Otherwise, a single botnet may control two comprehensive servers to perform a coordinated attack. When one server is operating, the other waits its turn. In addition, we found similar behavior in the principal component bases in Table 6, where the activities of downloading servers depend on bases. In phase 1, the set of severs in $\boldsymbol{u}_1$ are coordinated to perform downloads, while these servers are observed a totally different ratio in the phase 5. This is the reason why we claim some botnets were independent.

Note that botnet takeover phases are shown in some clusters of plot in the scatter plot Fig. 5. Phase 1 (decreasing line, $-4000 < y_1 < -1000$)) and phase 3 (and 5, increasing line, $500 < y_1 < 3000$) are isolated with totally different behaviors. This implies the set of active servers (IP addresses) in phases 1 and 3 are disjoint and hence can be considered as controlled in different botnets. Actually, the 1st and 2nd frequent servers have been used in downloading exclusively, i.e., the download from the 1st server broke when the 2nd server was active, and vice versa. The behavior of "negative correlation" between phases 1 and 3 is the evidence of our hypothesis that a botnet may intercept the existing server in competitive botnets from communicating.

3.8 Comparison to Other Botnet Detection Approaches

Botnet detection and tracking has been a major research topic and many attempts have been made. The simplest and the most common detection is the signature-based approaches used in commercial intrusion detection systems (IDS)[31]. Snort, an open source IDSs, can be used to detect known botnets based on the behavior of their communication. It is, however, not used for unknown botnets.

Botminer[30] is a system which applies clustering algorithm such as k-means to detect the Command and Control traffic. Botminer is an advanced detection tool which is independent from botnet protocols and structures, including IRC-based, HTTP-based and P2P structured botnets. However, there are two issues in detecting completing botnets; (1) the number of clusters has to be fixed before analysis. It is almost unpredictable. (2) the intermediate states between several phases are likely to be classified wrongly because the aggregated traffic from multiple botnets spoils the clear classifications.

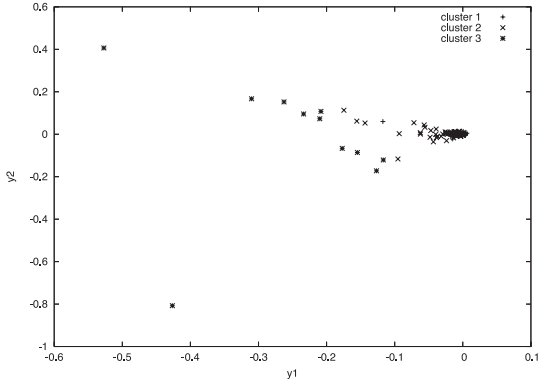

Fig. 11 Clusters of downloading server in k-means algorithm.

In order to compare our proposed PCA based detection with clustering approach, we apply the k-means algorithm to the same dataset used in our analysis and show the clustering results in **Fig. 11**, with $k = 3$ for matrix of 365 days $\times$ 100 addresses. The three clusters are plotted in the 2 dimensional graph with the first and the second principle components. Unfortunately, both two extreme servers (ranked 1 and 2 in Table 5) are classified into the same group, cluster 3 ($-0.6 < y_1 < -0.4$). This implies the clustering approach is not useful when there are some extremely frequent servers. Cluster 1 contains too many days to be joined in the same group.

On the other hand, our scheme uses PCA which allows us to detect multiple independent coordinated behaviors as orthogonal bases. Our experiments show there are multiple candidates of botnets and hence the aggregated traffic should be dealt with appropriately in a botnet detection system.

Consequently, we summarize our comparison of these major approaches in detecting botnets activities in **Table 8**.

Table 8 Comparison of Botnet detection techniques.

technique		unknown botnet	low false positive	aggregated botnes
Signature-based	Ref. 31)	–	√	–
Clustering	Ref. 30)	√	–	–
Our scheme (PCA)		√	√	√

4. Conclusions

We have discovered some useful features of the attacks made by coordinated servers. Our analysis reveals common features in the downloads of malware. We have identified interesting features of a coordinated attack performed via independent IP addresses.

This paper proposes a useful scheme for detecting and identifying coordinated botnet attacks. We aim to improve the accuracy of the infection estimation by introducing new features.

Acknowledgement

We thank Mr. Hiroshi Nakakoji, Mr. Tetsuro Kito, Mr. Masashi Fujiwara, and Mr. Tomohiro Kobori at Hitachi Ltd. for their useful suggestions.

References

1) Shadowserver: Botnet Charts, available from ⟨http://www.shadowserver.org/wiki/pmwiki.php/Stats/BotnetCharts⟩ (accessed 2009-09).
2) Octave, available from ⟨http://www.gnu.org/software/octave/⟩.
3) Terada, M., Takada, S. and Doi, N.: Network Worm Analysis System, *IPSJ Journal*, Vol.46, No.8, pp.2014–2024 (2005) (in Japanese).
4) Jung, J., Paxson, V., Berger, A.W. and Balakrishnan, H.: Fast Portscan Detection Using Sequential Hypothesis Testing, *Proc. 2004 IEEE Symposium on Security and Privacy* (*S&P'04*) (2004).
5) JPCERT/CC: ISDAS, available from ⟨http://www.jpcert.or.jp/isdas⟩.
6) Number of Hosts advertised in the DNS, Internet Domain Survey, available from ⟨http://www.isc.org/ops/reports/2005–07⟩ (accessed 2005-07).
7) Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S. and Weaver, N.: Inside the Slammer Worm, *IEEE Security & Privacy*, pp.33–39 (July 2003).
8) Shannon, C. and Moore, D.: The Spread of the Witty Worm, *IEEE Security & Privacy*, Vol.2, No.4, pp.46–50 (Aug. 2004).
9) Zou, C.C., Gong, W. and Towsley, D.: Code Red Worm Propagation Modeling and Analysis, *ACM CCS 2002* (Nov. 2002).
10) Moore, D., Shannon, C., Voelker, G. and Savage, S.: Network Telescopes: Technical Report, Cooperative Association for Internet Data Analysis (CAIDA) (July 2004).
11) Kumar, A., Paxson, V. and Weaver, N.: Exploiting Underlying Structure for Detailed Reconstruction of an Internet-scale Event, *ACM Internet Measurement Conference* (*IMC'05*), pp.351–364 (2005).
12) The Distributed Honeypot Project: Tools for Honeynets, available from ⟨http://www.lucidic.net⟩.
13) SANS Institute: Internet Storm Center, available from ⟨http://isc.sans.org⟩.
14) DShield.org: Distributed Intrusion Detection System, available from ⟨http://www.dshield.org⟩.
15) Kumar, A., Paxson, V. and Weaver, N.: Exploiting Underlying Structure for Detailed Reconstruction of an Internet-scale Event, *ACM Internet Measurement Conference* (2005).
16) Ishiguro, M. Suzuki, H., Murase, I. and Shinoda, Y.: Internet Threat Analysis Methods Based on Spatial and Temporal Features, *IPSJ Journal*, Vol.48, No.9, pp.3148–3162 (2007).
17) Dunlop, M., Gates, C. Wong, C. and Wang, C.: SWorD – A Simple Worm Detection Scheme, *OTM Confederated International Conferences: Information Security* (*IS 2007*), LNCS 4804, pp.1752–1769 (2007).
18) JPCERT/CC: Increased activity targeting TCP port 5168, JPCERT-AT-2007-0019 (2007), available from ⟨http://www.jpcert.or.jp/at/2007/at070019.txt⟩.
19) Ishiguro, M., et al.: Feature Analysis of Illegitimate Packets Monitored on the Internet, *IPSJ Computer Security Symposium* (*CSS 2005*) (2005).
20) Hatada, M., et al.: Malware Workshop and Common Data set, *IPSJ Malware Workshop* (*MWS2008*) (2008).
21) Fujiwara, M., et al.: Malware Analysis and Classifications, IPSJ Technical Report, pp.177–182 (2008).
22) Kikuchi, H., Matsuo, S. and Terada, M.: Principal Component Analysis of Botnets Takeover, *The 5th Joint Workshop on Information Security* (*JWIS 2010*) (2010).
23) Lu, W., Tavallaee, M. and Ghorbani, A.A.: Automatic Discovery of Botnet Communities on Large-Scale Communication Networks, *ACM ASIACCS'09* (2009).
24) Network World: Russian botnet tries to kill rival (2010), available from ⟨http://www.networkworld.com/news/2010/021010-new-russian-botnet-tries-to.html⟩.
25) Barford, P. and Yegneswaran, V.: An inside look at Botnets, Special Workshop on Malware Detection, Advances in Information Security, Springer Verlag, ISBN: 0-387-32720-7 (2006).
26) Strayer, T., Walsh, R., Livadas, C. and Lapsley, D.: Detecting botnets with tight command and control, *Proc. 31st IEEE Conference on Local Computer Networks*, pp.195–202 (2006).
27) Gu, G.F., Zhang, J.J. and Lee, W.K.: BotSniffer: Detecting botnet command and

control channels in network traffic, *Proc. 15th Annual Network and Distributed System Security Symposium*, San Diego, CA (Feb. 2008).
28) Kuwabara, K., Kikuchi, H., Terada, M. and Fujiwara, M.: Heuristics for Detecting Botnet Coordinated Attacks, *International Conference on Availability, Reliability and Security* (*ARES 2010*), *WAIS 2010*, pp.603–607, IEEE (2010).
29) Kozakura, F., Tsuda, H. and Torii, S.: Visualizing Malware Behavior with Business Information Navigator, *IPSJ Magazine*, Vol.51, No.3, pp.288–292 (2010) (in Japanese).
30) Gu, G., Perdisci, R., Zhang, J. and Lee, W.: BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection, *Proc. 17th USENIX Security Symposium* (2008).
31) Snort IDS (2009), available from ⟨http://www.snort.org⟩.
32) Feily, M., Shahrestani, A. and Ramadass, S.: A Survey of Botnet and Botnet Detection, *3rd International Conference on Emerging Security Information, Systems and Technologies*, (*SECURWARE '09*), pp.268–273, IEEE (2009).

(Received November 30, 2010)
(Accepted June 3, 2011)
(Original version of this article can be found in the Journal of Information Processing Vol.19, pp.463–472.)

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 until 1993, he joined Tokai University in 1994. He is currently a Professor at 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, cryptographical protocol, and network security. He is a member of the Institute of Electronics, Information and Communication Engineers of Japan (IEICE), the Japan Society for Fuzzy Theory and Systems (SOFT), IEEE and ACM. He is a fellow of the Information Processing Society of Japan (IPSJ).

Shuji Matsuo was born in Japan. He received his B.E. and M.E. degrees from Tokai University in 2008 and 2010. He has been working for Fujitsu Ltd., Network Solution division since 2010. His research interests are in Network security and network applications.

Masato Terada was born in Japan. He received the 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 the 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 the Information Processing Society of Japan (IPSJ).