Heuristics for Detecting Botnet Coordinated Attacks

Kazuya Kuwabara
Hiroaki Kikuchi
Graduate School of Science and Technology,
Tokai University,
1117 Kitakaname, Hiratsuka, Kanagawa 259-1292, Japan
mulberry@cs.dm.u-tokai.ac.jp
kikn@cs.dm.u-tokai.ac.jp

Masato Terada
Masashi Fujiwara
Hitachi Incident Response Team (HIRT),
Hitachi, Ltd.
890 Kashimada, Kawasaki, Kanagawa, 212-8567, Japan
masato.terada.rd@hitachi.com
masashi.fujiwara.zz@hitachi.com

***Abstract*— This paper studies the analysis on the Cyber Clean Center (CCC) Data Set 2009, consisting of raw packets captured more than 90 independent honeypots, in order for detecting behavior of downloads and the port-scans. The analyses show that some new features of the coordinated attacks performed by Botnet, e.g., some particular strings contained in packets in downloading malwares, and the common patterns in downloading malwares from distributed servers.**

Based on the analysis, the paper proposes the heuristic techniques for detection of malwares made by Botnet coordinated attack and reports the accuracy of the proposed heuristics. The detection process is automated in the proposed decision tree consisting of statistics, such as, a number of total inbound packets, and an average rate of downloading malwares.

I. Introduction

Botnet is a set of malicious software robots running distributed environments, under the control of bonet's originator, called "herder" or "bot master". The set of compromised hosts jointly perform attacks for looking for vulnerabilities in target network. These attempts are usually made on a specific destination port for which services with known vulnerable software are available. Ports 135, 138, and 445 are frequently scanned. There is also malicious software that uses particular ports to provide a "back door" to companies.

In this paper, we study the CCC (Cyber Clean Center) DATA set 2009, consisting of raw packet data captured more than 90 independent honey pots for two years. Our purpose is to clarify typical behavior of botnet from the observation of CCC DATA set. The characteristics are useful for heuristic method for detecting and predicting botnet coordinated attacks. The CCC, the Japanese governmental organization, is observing the backbone of Japanese tier-1 providers. Honey pot is a virtual host running two guests OSs, periodically rebooted. The CCC DATA set 2009 provides 145 time slots, a period of time between reboots of honeypot.

According to recent report in [3], multiple servers in a botnet collaborate to attack a single vulnerable hosts under the control of the botnet master. Figure 1 illustrates a typical time chart how three servers S1, S2, and S3 coordinate to attack a target host, where S1 keeps sending malware classified as PE at time t0 through t1, S2 and S3 send malwares TROJ and WORM, respectively. The infected host is controlled by botnet via IRC server, by receiving IRC message, e.g., NICK or JOIN. ΔT_2 second after the host gets the JOIN message, the infected host performs port-scans to specified destination network. As shown in the Fig. 1, the single host is attacked by several coordinated servers so that the botnet prevents from detecting their servers. Hence, in this paper we propose several heuristics for detecting botnet behavior in advance based on the observation of the CCC DATA set. Finally, we show the accuracy of the proposed technique using the experimental data.

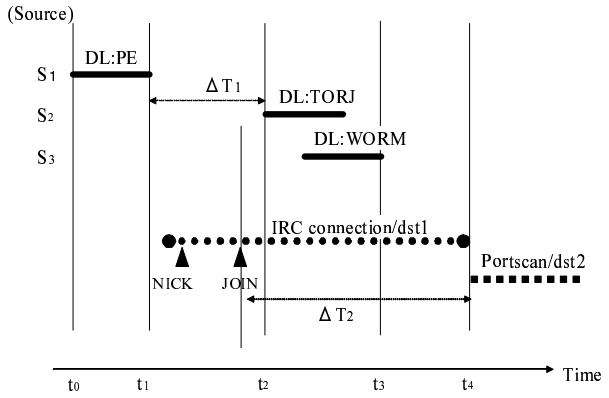

Fig. 1. Time chart of coordinated attack

II. Botnet Analysis

A. Features of Botnet

The botnet sends packets containing particular messages, e.g., `MZ`, `PE` and `DOS`. The typical executable binary has a message "`!This program cannot be run in DOS mode.`", or "`!Windows Program`" appears when the file in used tftp is downloaded. The command-and-control (C&C) server may send a message contains `NICK` or `JOIN`, which are used in the IRC protocol. The `NICK` is a command when nickname is applied, `JOIN` is a command to enter the channel. The `Ipscan` is for port-scanning. We identify the type of the portscans into fourth octet changed one by one, referred as s_4. Table I shows a list of features for classification of botnet messages.

TABLE I
LIST OF FEATURES

symbol	features
$slot$	slot ID (0, ..., 145)
$Time$	Beginning time in slot
P_I, P_O	In bound & Out bound [pkt]
MZ	Appears character string named "MZ"
PE	Appears character string named "PE"
DOS	Appears character string named "`!This program cannot be run in DOS mode.`"
win	Appears character string named "`!Windows Program`"
N, J	Appears character string named "NICK"&"JOIN"
$ip1$	Appears character string named "``#las6 * ipscan s.s.s.s dcom2 -s`"
$ip2$	Appears character string named "``#last * ipscan s.s.s.s dcom2 -s`"
ST	port scan type
DL	counts of downloading
MW	name of malwares

B. List of Malwares

Our analysis shows 24 unique hash values from total of 200 values, listed in Table II. The 24 kinds of unique hash values are identified 13 kinds of unique malware names.

C. Examination of detection communication in botnet

There are several useful tools of analysis including snort[4], BotHunter[5] and BotSniffer[6]. The snort compares the extracted host name with the blacklist. BotHunter recognizes the communication patterns of computer infected by malware, and discover traffic in bot. BotSniffer is a system that detects C&C of the botnet. The correlation among C&C servers and the characteristic of the similarity are used. The BotHunter classifier the clients into the groups according to the address of IP and the port number and to examine the correction of time and the space.

III. ANALYTICAL RESULT

Table III shows List of the features of Time slots. We find that 58 slots infected by the malware out of 145 slots, and 6 slots in infection by using tftp and UDP.

The malware BKDR_RBOT.ASA is observed in five slots. There is a single slot that has BKDR_MYBOT.AH. The infection with tftp/UDP does not have the string "`This program cannot be run in DOS mode.`" with appears often in downloading with TCP.

Figure 2 shows the number of output packets downloaded by TFTP. When the malware is downloaded in UDP, it takes longer time than that of in TCP.

A. Heuristics for Detecting Portscan

We discovered heuristics for detecting portscans. Figure 3 shows the time difference from sending the command "JOIN" to the time to perform the port-scan. The horizontal axis indicates time of JOIN, the vertical axis shows the time of the port-scan.

The type of the port scan in this case was classified as s4, i.e., increasing the fourth octet of IP address. The heuristics in terms of port-scan are as follows.

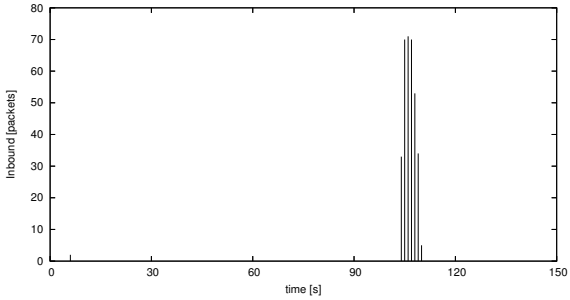

Fig. 2. Inbound packets (UDP)

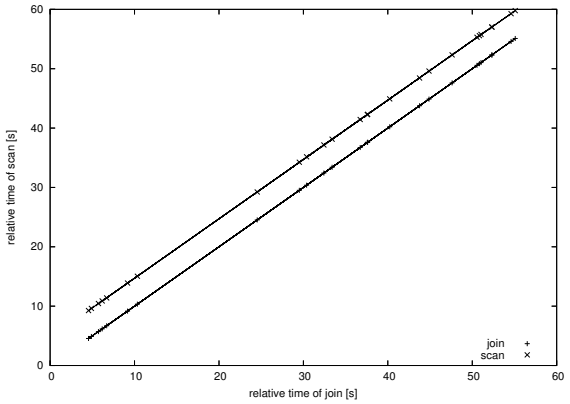

Fig. 3. Distribution of time difference between JOIN and portscan

R 1a. Port-scan is performed after five seconds it received JOIN command.
R 1b. Port-scaning host sends 256 packets per a second.
R 1c. PE_VIRUT.AV scans destination addresses with 1st and 2nd octect unchanged.

B. Heuristic for detecting coordinated infections

There are 58 slots to download the malware out of 145 slots. Table IV shows three commonly shown patterns of coordinated

TABLE II

LIST OF MALWARES AND ITS 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

TABLE III

TABLE OF FEATURES FOR TIME SLOT

$slot$	P_I	P_O	MZ	PE	DOS	$NICK/JOIN$	$ip1/ip2$	$ST(s4)$	$infection$	$malwares$
0	276	17774	9	13	3	1		1	1	$PE1$, $TR2$, $WO3$
1	61	352	0	4	0				0	
2	7488	178491	10	16	3	1	$ip2 \times 1$	1	1	$WO1$, $PE1$, $TR2$, $WO3$
3	350	240148	12	10	4	1	$ip2 \times 1$	1	1	$PE1$, $TR2$, $WO3$, $PE1$
4	2	55	0	0	0				0	
5	5	59	0	0	0				0	
14	354	135725	9	10	3	1	$ip1 \times 3$	1	1	$BK1$, $TR2$, $WO3$
55	822	179581	21	16	7	1	$ip1 \times 2$	1	1	$BK1$, $WO3$, $TR2$, $BK1 \times 4$
46	379	791	0	0	0				1	$BK2$
83	571	74286	15	15	5	1		1	1	$PE1 \times 2$, $TR2$, $WO3$
139	450	96211	13	18	3	1	$ip2 \times 1$	1	1	$PE2$, $WO4$, $WO3$
140	691	101877	21	24	5	1	$ip2 \times 1$	1	1	$PE2$, $WO4$, $WO3$
total	44452	3038276	691	966	219	60	33	28	58	200
ave	306.57	20953.63	4.77	6.66	1.51	0.41	0.23	0.19	0.4	1.38

infections.

R 2a. WORM_SWTYMLAI.CD and TROJ_BUZUS.AGB downloaded at the same time after PE_VIRUT.AV is downloaded .

R 2b. Source IP address of WORM_SWTYMLAI.CD and TROJ_BUZUS.AGB are identical.

R 2c. WORM_SWTYMLAI.CD and TROJ_BUZUS.AGB use the port number of 80 and PE_VIRUT.AV uses port numbers of five digits long.

Rules from R 2a through R 2c are particular features in the coordinate infections.

Table IV shows top three commonly shown patterns of coordinated infections. All coordinated infections are classified into the three patterns in the table. The most frequent pattern, R 2a, is observed in 17 time slots out of 58 infected slots.

Table V shows in detail of packets communicated in the most common coordinated infection pattern. From the observation, we find the features in terms of IP addresses and destination port numbers in Rule 2b and 2c.

Table VI shows the number of distinct downloading servers used in the 1st pattern of coordinated infection. The

TABLE IV

PATTERNS OF COORDINATED INFECTIONS

Pattern	number of time slots
PE1 → TR2, WO3	17/58
BK1 → TR2, WO3	5/58
PE2 → WO4, WO3	4/58

TABLE VI

NUMBER OF DISTINCT DL SERVERS

MW	distinct DL servers
PE_VIRUT.AV	10
TROJ_BUZUS.AGB	1
WORM_SWTYMILAI.CD	1

PE_VIRUT is distributed from 10 distinct servers, while the subsequent malwares are particular addresses.

Table VII shows the relationship of IP addresses assigned for the downloading server (adversary), the honeypot, and the destination network to which the honeypot performs port-scanning. We find the common feature described in Rule R 2d.

TABLE V

EXAMPLES OF COORDINATED INFECTIONS

$slot$	$time$	$srcIP$	$dstPort$	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
3	0:46:56	124.86.A2.B2	33258	PE_VIRUT.AV
3	0:48:52	67.215.C1.D1	80	TROJ_BUZUS.AGB
3	0:48:52	72.10.E1.F1	80	WORM_SWTYMLAI.CD
16	5:17:25	114.145.A3.B3	15224	PE_VIRUT.AV
16	5:18:37	67.215.C1.D1	80	TROJ_BUZUS.AGB
16	5:18:38	72.10.E1.F1	80	WORM_SWTYMLAI.CD

TABLE VII

IP ADDRESSES ASSIGNMENTS

$slot$	DL server	honey pot	destination of scans
0	$124.86.C1.D1$	$124.86.E1.F1$	$124.86.E1.F1+1$
2	$124.86.C2.D2$	$124.86.E2.F2$	$124.86.E2.F2+1$
3	$124.86.C2.D2$	$124.86.E2.F2$	$124.86.E2.F2+1$
16	$114.145.C3.D3$	$114.145.E3.F3$	$114.145.E3.F3+1$
29	$114.164.C4.D4$	$114.164.E4.F4$	$114.164.E4.F4+1$
	$A.B.C.D$	$A.B.E.F$	$A.B.E.F+1$

Table VIII shows the statistics of three coordinated infections including the number of slots, the mean duration of downloading, and the standard deviation

C. Heuristic for identiting kinds of malwares

There are two kinds of the ways of connection, PULL and PUSH. PULL requires the honeypot to initiate the connection to the downloading servers. PUSH makes honeypot to listen connection at a port specified by the C&C server, and waits for the malware to open connection from the download host. The behaviors in downloading malwares depend on the kind of malware. Rules in 3a through 3d are heuristics in terms of downloading. Rule 3b and 3d describe particular strings used in downloading executable malwares.

Figure 4 shows the number of inbound packets in a time slot when malware is downloaded in PUSH-style connection. The sending packets rate varies for PUSH and PULLS connections.

- R 3a. The downloading in PUSH sends packets in constant rate.
- R 3b. Packets containg string, "MZ" and "PE" use TCP to download malwares.
- R 3c. The downloading in PUSH is made by WORM_ALL APLE.
- R 3d. Downloading in TFTP, contains string "win" in UDP.

D. Algorithm for identification of infection

We propose an algorithm to identity kinds of infections shown in figure 5, which classifies time slot based on the features Rules of a heuristic.

The tree begins classifing slot be testing the total output packets are more than 85 packets or not. The strings "MZ", "PE", "DOS", and "Win" are used to classify slots. The tree uses CCC DataSet 2009 as learning data and is tested with the same dataset. The accuracy of this algorithm is summarized in Table IX. It shows no false detection is made.

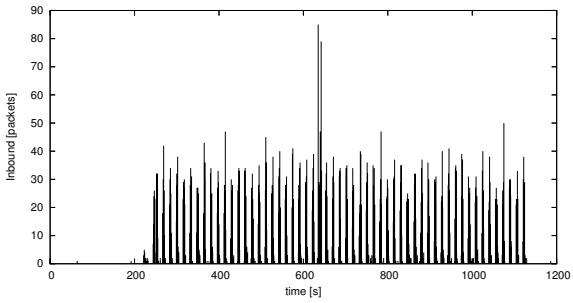

Fig. 4. Inbound packet (PUSH)

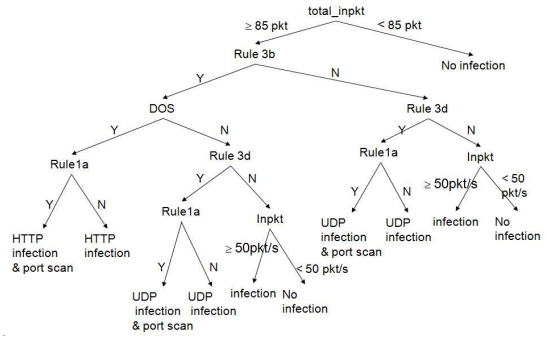

Fig. 5. Algorithm for detecting infections

E. Determined by the decision tree

Using a typical decision tree algorithm C4.5[23], decision tree diagram to determine the infection was shown to extract Figure 6. Compared to Figure 5, the number of nodes less one of four that have been trying to optimize, out _ pkt < 338 are classified in seven slots, erroneous decision of infection (False Positive) is one of the slots have caused.

TABLE VIII

STATISTICS OF COORDINATED INFECTIONS

		slot	# of slots	mean duration	Std.deviation
pattern 1	PE1⇒ TR2, WO3	0, 2, 3, 16, 29, 30, 50, $\cdots$	17	127.24	158.75
pattern 2	BK1⇒ TR2, WO3	14, 55, 56, 125, 126	5	176.4	147.36
pattern 3	PE2⇒ WO4, WO3	66, 139, 140, 141	4	253.25	176.25

TABLE IX

ACCURACY OF HUERISTICS FOR DETECTING INFECTION

result \ true	infection	not infection	total slot
infection	58	0	58
not infection	0	87	87
total slot	58	87	145

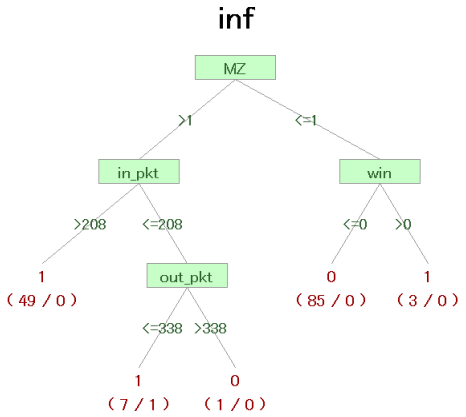

Fig. 6. Decision Tree for Detecting Infection using C4.5

IV. Conclusion

We discovered some useful features in terms of the attacks made by some coordinated servers. Our analysis shows some common features in the downloading of the malwares. We have shown an interesting features of the coordinated attack performed independent IP addresses.

This paper gives several useful heuristics for detecting and identifying botnet attacks. We will improve the accuracy of the infection estimation introduceing by new features.

Acknowledgement

We are greatful to Mr. Hiroshi Nakakoji and Mr. Tetsuro Kito at Hitachi Ltd. for their useful advice.

References

[1] Hatada, et. al, "Malware Workshop and Common Data set", IPSJ Malware Workshop (MWS2008), 2008.

[2] Kobori, Kikuchi and Terada, "Correlation between Observation and Portscanning Attacks", IPSJ Malware Workshop (MWS2008), pp.67-74, 2008.

[3] Fujiwara et. al, "Malware Analysis and Classifications", IPSJ Technical Report, pp. 177-182, 2008.

[4] snort, `http://www.snort.org/` (See 2009 Mon 11).

[5] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, and Wenke Lee, "BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation," USENIX, Proc. of 16th USENIX Security Symposium, 2007.

[6] Guofei Gu, Junjie Zhang, and Wenke Lee, "Botsniffer:Detecting botnet command and control channel, " Internet Society, Proc. of NDSS 2008, Feb. 2008.

[7] Y.Higashikado, S.Torii, "Study for Detecting Bot-Infected PC based on behavior of DNS query", MWS2008, pp. 13-18, 2008 (in Japanese)

[8] M. Terada, S. Takada, and N. Doi, "Network Worm Analysis System", *IPSJ Journal*, Vol. 46, No. 8, pp. 2014–2024, 2005 (in Japanese).

[9] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast Portscan Detection Using Sequential Hypothesis Testing", *Proc. of the 2004 IEEE Symposium on Security and Privacy (S&P'04)*, 2004.

[10] JPCERT/CC,ISDAS, (http://www.jpcert.or.jp/isdas)

[11] "Number of Hosts advertised in the DNS", Internet Domain Survey, July 2005. (http://www.isc.org/ops/reports/2005–07).

[12] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the Slammer Worm", *IEEE Security & Privacy*, pp. 33–39, July 2003.

[13] C. Shannon and D. Moore, "The Spread of the Witty Worm", *IEEE Security & Privacy*, 2(4), pp. 46–50, August 2004.

[14] C. Changchun Zou, W. Gong, and D. Towsley, "Code Red Worm Propagation Modeling and Analysis", *ACM CCS 2002*, November 2002.

[15] D. Moore, C. Shannon, G. Voelker, and S. Savage, "Network Telescopes: Technical Report", Cooperative Association for Internet Data Analysis (CAIDA), July 2004.

[16] A. Kumar, V. Paxson, and N. Weaver, "Exploiting Underlying Structure for Detailed Reconstruction of an Internet-scale Event", *ACM Internet Measurement Conference (IMC 05)*, pp. 351–364, 2005.

[17] The Distributed Honeypot Project: "Tools for Honeynets." (http://www.lucidic.net).

[18] SANS Institute: "Internet Storm Center", (http://isc.sans.org).

[19] DShield.org, "Distributed Intrusion Detection System", (http://www.dshield.org).

[20] A. Kumar, V. Paxson, and N. Weaver, "Exploiting Underlying Structure for Detailed Reconstruction of an Internet-scale Event", ACM Internet Measurement Conference, 2005.

[21] Masaki Ishiguro, Hironobu Suzuki, Ichiro Murase and Yoichi Shinoda, "Internet Threat Analysis Methods Based on Spatial and Temporal Features", *IPSJ Journal*, Vol. 48, No. 9, pp. 3148–3162, 2007.

[22] Matthew Dunlop, Carrie Gates, Cynthia Wong, and Chenxi Wang, "SWorD – A Simple Worm Detection Scheme", *OTM Confederated International Conferences: Information Security (IS 2007)*, LNCS 4804, pp. 1752–1769, 2007.

[23] Quinlan, J. R., , "C4.5 Progarams for Machine Learning", Morgan Kaufmann, San Mateo, California.