Journal of Computers Vol. 29 No. 5, 2018, pp. 190-201
doi:10.3966/199115992018102905015

Detection of C&C Servers Based on Swarm Intelligence Approach

Chia-Mei Chen[1*], Gu-Hsin Lai[2], Jiunnwu Lin[3]

[1] Department of Information Management, National Sun Yat-Sen University, Kaohsiung 804, Taiwan
cchen@mail.nsysu.edu.tw

[2] Department of Technology Crime Investigation, Taiwan Police College, Taipei 116, Taiwan

Received 7 January 2018; Revised 7 July 2018; Accepted 7 August 2018

Abstract. Botnets have caused significant security threat and huge loss to organizations. It is difficult to discover their existence and botnet attacks are one of the biggest challenges that security administrators face today. Bots are controlled by command and control servers (C2 servers) and used to conduct cyber warfare performance various attacks. To improve survivability and evade intrusion detection, an advanced botnet may consist of multiple command and control servers. Therefore, identifying C2 servers is important to prevent botnet attacks or further damage. This study proposes a distributed detection scheme based on ant colony optimization algorithm which discovers the paths from bots to C2 servers. The results show that the proposed detection can identify botnet servers efficiently.

Keywords: anomaly detection, ant colony optimization, Botnet

1 Introduction

The security incident reports [1] discovered that the attackers often controlled a botnet to perform malicious actions, while the real attackers are behind the scene. Botnet size may vary ranging from thousands to hundred millions of bots [2-3] and the power of a botnet is proportional to the size. Therefore, detecting botnets is of great importance.

Command and control servers play a vital role in botnets; taking down the servers means disrupting the attacking power of a botnet. For the purpose of robustness and intrusion evasion, botmasters may build multiple command and control servers which communicate with bots by using a commonly used network protocol and hide malicious transmissions in a vast amount of normal user traffic.

Advanced botnets adopt fast-flux domain technology to extend the lifetime and survivability [4-6]. Based on the structure of fast-flux domain as shown in Fig. 1, fast-flux agents act as a relay station between the command and control servers and the clients. Fast-flux domains achieve stealthy by mapping to different domains and IP addresses dynamically. As the bots connects one of the command and control servers dynamically, it makes botnet detection more challenging.

Even though the malicious traffic is small, the communication exhibits certain anomaly behaviors as a bot is robot software. Normal user requests are issued at a random time and the contents are diverse, while bots may connect to the server periodically and the message content may be limited. This study proposes a botnet server detection mechanism based on ant colony optimization. This study proposes a new pheromone formula which transforms the idea of finding shortest path to finding anomaly traffic flow. Therefore, the ant colony optimization algorithm can be adopted to find the anomalous traffic between bots and the command and control servers and discover the servers.

[*] Corresponding Author

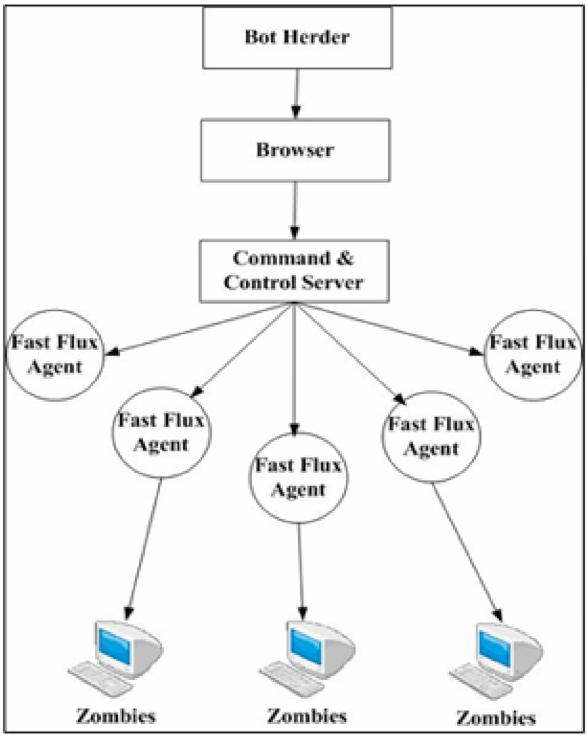


Fig. 1. Botnet with multiple command and control servers

2 Related Work

A botnet detection system BotGAD (Botnet Group Activity Detector) [7] is developed, based on the group activity model and metric, including group uniformity, activity periodicity, and activity intensity. Lakhina et al. [8] adopted sample entropy to find the traffic flow distribution characteristics. The detection approach can detect various attacks such as DDoS and port scan during the progress of the attacks, but it is not able to identify botnets prior to the attacks.

AsSadhan et al. [9] employed periodograms to study the periodic behavior of botnet and monitor the command and control communication traffic. The Walker's large sample test is applied to detect the C&C traffic whether bot traffic is or not. Yen and Reiter [10] proposed a detection system called TAMD to identify infected hosts in the enterprise network by finding out aggregated communication involving multiple internal hosts. The aggregated features include flows communicating with the same external network, sharing similar payload, and involving internal hosts with similar software platforms. The experimental results show that the proposed approach has a low false positive rate.

A bot is a program which can perform a fixed number of instructions. All bots commit malicious activities according to the botmaster's commands. Akiyama et al. [11] proposed three metrics for determining the botnet behaviors: relationship, response, and synchronization. The relationship presents the connection between botmaster and bots over one protocol, such as IRC, HTTP, or P2P. The response means that bots respond immediately and accurately after they receive commands from the botmaster. The synchronization means bots simultaneously carry out programmed activities, such as DDoS attack, reporting their status, or sharing information, based on the botmaster's commands.

Some studies employed data mining technique as countermeasure for botnets. Livadas et al. [12] and Strayer et al. [13] apply machine learning algorithms, like C4.5 Decision Tree, Naïve Bayes, and Bayesian Networks, to analyze IRC-based botnet. The network traffic is classified into two groups: IRC and non-IRC. The Euclidean distance is calculated to correlate similar IRC traffic together. Kondo and Sato [14] adopted support vector machine (SVM) algorithm to identify C&C sessions from the traffic data. Their work finds out the packet histogram vector of the C&C session, including packet payload size and packet interval time, can better identify C&C sessions than the other vector definitions, such as session information vector and packet sequence vector. Lu et al. [15] employed n-gram, decision tree and clustering algorithms to classify network traffic into different application communities. Lu's work analyzes the temporal-frequent characteristics of the 256 ASCII bytes on the payload over a predefined time interval to distinguish malicious bot traffic from normal one. Huang [16] proposed a bot detection mechanism which analyzed failure packets by means of machine learning approach.

ACO has been applied to DoS attack traceback [17], as the attacker might use spoofed source IP addresses to launch an attack. A path link whose traffic fulfills DoS attack traffic accumulates more pheromone and ants explore the path with more pheromone. DoS attack path will be identified after the evolution of the ants. Inspired by the previous work, Wang et al. [18] proposed botnet attack path finding based on ACO. The discovery scheme requires complete or partial traffic information of the whole network, while most network administrators only possess the traffic information of its own administrated network.

Chen and Huang [19] proposed an ACO-based detection framework to identify low-rate DDoS attacks. The framework comprises three stages and ACO algorithm is applied to collect information heuristic from the network traffic. Castiglione et al. [20] proposed a new botnet-based command and control approach which relies on ACO to improve the survivability and scalability of botnets. The authors claimed that this might be a new evolutionary malware-based control scheme in the future.

ACO has been applied to shortest path routing, traveling salesperson, and optimal network routing problems [21-22]. The network routing research [23] demonstrated that ACO performs better than others and introduced two types of ants for changing routing cost: regular ant and uniform ant. The improved ACO is more suitable for finding an optimal routing in dynamic network environments.

The above research inspired us to apply ant colony optimization algorithm to identify C&C servers by leaving more pheromone on anomalous traffic flows. Therefore, the ant colony optimization algorithm can be adopted to find the anomalous traffic between bots and the C&C servers.

3 Proposed Detection Approach

We observe that the change of the pheromone attempts to search for short edge and then induce a positive feedback. If the short edges are transformed into the regularity found in the communication between the bot machines and the C&C servers. Normal client and server communication does not have regular connection frequency or the similar payload, while, to keep the bots and the C&C server connected, the bots exhibit certain regularity on the connectivity and the contents. This study discovers the following attributes about the communication between the bots and the C&C servers in a time frame: (1) regular communication period, (2) anomaly messages, and (3) similar payload.

To identify abnormal botnet traffic among a vast amount of network traffic, the proposed detection examines the outbound network flow information of a corporate network. Ants explore the traffic in a time frame and tend to choose the path with the anomalous traffic. The traffic to a destination will accumulate more pheromone, if it exhibits the three attributes described above. Ants continue exploring all the traffic until optimal path is found and the C&C server is identified.

As mentioned above, a bot is a program which can perform a fixed number of instructions. The communication between bots and C&C server exhibits some regularity on the content and frequency, while the normal traffic usually contains various message contents and random connection time and periods. This study adopts anomaly score to determine the pheromone. The anomaly score computes the degree of the regularity found in one connection so that ants tend to explore the paths with regular connection attributes and to identify the communications between bots and C&C server.

The system architecture of the proposed ACO-based detection is plotted in Fig. 2. Firstly, the module Traffic Preprocess collects and organizes traffic flows into a set of traffic packets by a time frame. The module Feature Selection extracts the features from the payloads and computes the anomaly score of each path in a given time frame. A connection exhibiting regularity on payload contents and frequency might be one between bot and its server. The ant algorithm chooses the paths to explore according to the anomaly scores. Once all the traffic has been explored by the ants, the list of suspicious C&C servers is reported.

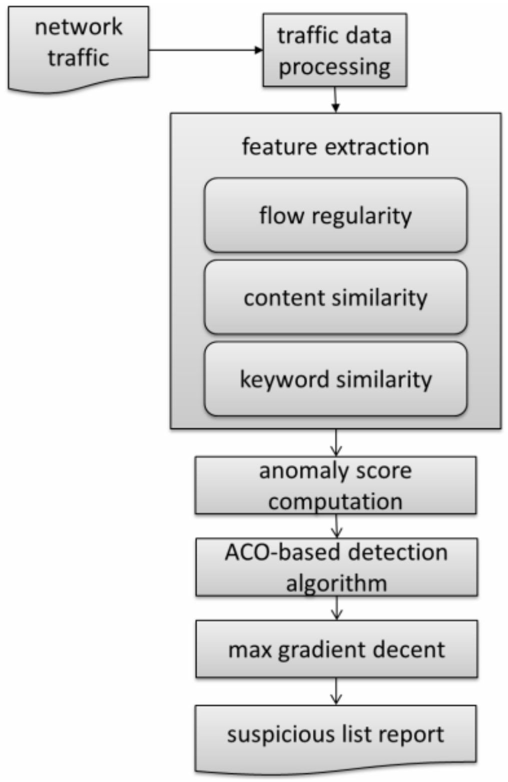


Fig. 2. The proposed system architecture

3.1 Preprocess

An IRC traffic logger, IRC sniffer, is deployed to collect IRC traffic flows in the proposed system. The flow features are selected for further analysis as shown in Table 1. The source and destination IP address are extracted from the IP header and the source and destination port are extracted from TCP header. The payload information is extracted from payload of IRC traffic including IRC commands such as JOIN, USER, PASS and IRC messages embedded in PRIVMSG. In this paper, the basic analysis unit, a flow, is defined as 6-tuple R = {Sip, Dip, Sport, Dport, Time, Payload}.

Table 1. Flow Features

Feature name	Description
Sip	Source IP Address
Dip	Destination IP Address
Sport	Source Port
Dport	Destination Port
Time	Timestamp
Payload	Payload of the connection

3.2 Feature Selection and Anomaly Score

This study adopts three attributes to define the anomalous communication between bots and its server: flow regularity, content similarity, and keyword similarity. The following notations will be used for computing the attributes. Let $CR(Sip_a, all) = CR_a$ be the set of traffic flows from Sip_a, $CR(Sip_a, Dip_b) = CR_{ab}$ be the set of traffic flows from Sip_a to Dip_b, and $R_i(Sip_a, Dip_b)$ be the ith flow from Sip_a to Dip_b. $|CR_{ab}|$ denotes the number of traffic flows in CR_{ab} and $|R_i|$ denotes the size of flow R_i.

The attribute flow regularity contains three indices: S_t: the standard deviation of the interleave time of two consecutive connections, S_s: the standard deviation of the packet size, and h: the ratio of the number of traffic flows destined to Dip_b to the total number of flows in the traffic cluster $CR(Sip_a, all)$. Bots would communicate with the servers intensively during the active time; therefore, the traffic deviations become small.

For *CR(Sip$_a$, Dip$_b$)*, the cluster of traffic between the two parties *Sip$_a$ and Dip$_b$*, a small standard deviation of the interleave time means the two parties have regular connections and a small standard deviation of the packet size implies the messages are similar. If the cluster contains a large portion of traffic to a specific destination (i.e., *h* is high), it means the communication between the two parties is intensive. The first attribute, flow regularity, *f$_1$(Sip$_a$, Dip$_b$)* is defined as follows.

$$f_1(Sip_a, Dip_b) = \frac{\sqrt{1/(S_t+1)} + \sqrt{1/(S_s+1)} + h}{|CR_{ab}|^2} \quad .$$

Some peer-to-peer traffic might possess the flow regularity attribute. Therefore, further attributes are needed to distinguish the difference. For example, for IRC-based botnet, the bots and the server usually will exchange message of connection alive periodically. For web-based botnet, the bots might regularly check the webpage content. The second attribute, content similarity, compares the message content similarity by longest common subsequence (LCS) and averages the degree of the similarity of all traffic flows in a given time frame. The attribute formula *f$_2$(Sip$_a$, Dip$_b$)* is shown as follows.

$$f_2(Sip_a, Dip_b) = \frac{\sum_{all(Ri,Rj)} Sim(R_i, R_j)}{(|CR_{ab}|*(|CR_{ab}|-1))/2},$$

where |CR$_{ab}$| denotes the number of traffic flows in *CR(Sip$_a$, Dip$_b$)* and the similarity of two message contents is defined below.

$$Sim(R_i, R_j) = \frac{LCS(R_i, R_j)}{(|R_i|+|R_j|)/2} \text{ for } R_i,\ R_j \in \mathrm{C}R_{ab}.$$

The third attribute identifies the suspicious keywords in the messages. As a bot machine is not a human, it understands a limited set of commands or words. Therefore, a bot might receive or response similar messages during the communication with the C&C server. A web-based bot machine usually uses GET/POST to communicate with its server and might appear GET*.exe or POST*.exe to download the update. An IRC-based one might appear keywords, such as USER, NICK, PING, PONG, MODE, and JOIN. Some PRIVMSG message may contain attack keywords issued by botmaster, such as DDOS, FLOOD, and INFO. Therefore, the third attribute, *f$_3$(Sip$_a$, Dip$_b$)*, keyword similarity, computes the average ratio of the number of keywords appeared in the messages between the two parties and the formula is defined below.

$$f_3(Sip_a, Dip_b) = \sum_{allRi} \frac{Suspected\_Token(R_i)}{TotalToken(R_i)} / |CR_{ab}|.$$

In total, the anomaly score function, *F(Sip$_a$, Dip$_b$)*, considers the three attributes with different weights. That is, $F\ (Sip_a, Sip_b) = F_{ab} = w_1 f_1 + w_2 f_2 + W_3 f_3$, where the summation of the three weights equals 1.

Let N be the maximum number of connections a host would make in any time and *CR* be the maximum number of flows in a pair of source and destination. The time complexity of computing the first attribute is *O(CR*N)*; that of the second attribute is *O(CR^2);* that of the third attribute is *O(CR)*. It yields that the time complexity of feature extraction and anomaly score computation is *O(CR^2)*.

3.3 Detection Algorithm

An isolated ant moves essentially random. It decides to follow a trail with high pheromone trail and reinforces the trail by laying its own pheromone. The collective behavior emerging from ants is a form of autocatalytic reaction where the more the ants follow a trail, the more attractive the trail becomes. The proposed ACO-based detection algorithm develops metaheuristic information, anomaly score function, which signifies the immediate impact that a local decision might have on solution quality. In this study, traffic path exhibiting bot-server connection behaviors has high pheromone and more ants will explore such path. If the same traffic path continues showing such anomaly, ACO will form a positive feedback and finally most ants will explore the same path.

In the initialization phase, a group of ants is positioned on a client machine in the network. The cluster of traffic flows from the client Sip_a in a given time frame, i.e., $CR(Sip_a, all) = CR_a$ is examined and the pheromone is defined below.

$$P_{ab}(t)=\frac{[\tau_{ab}(t)]^{\alpha}\times[\eta_{ab}(t)]^{\beta}}{\sum_{R_{ak}\in CR_a}[\tau_{ak}(t)]^{\alpha}\times[\eta_{ak}(t)]^{\beta}},$$

where $\tau_{ab}(t)$ is the intensity of the pheromone on the trail *(Sip$_a$, Dip$_b$)* at time frame *t*, $\eta_{ab}(t)$ is visibility function in ACO algorithm and is defined as the anomaly score function *F* in this study, and Dip_k is *k*th destination IP in the cluster CR_a. α and β balance the impact of the two factors, pheromone and visibility, and α + β = 1.

In the pheromone calculation, visibility function, η, is often defined as the reciprocal of distance, where the shorter distance contributes larger visibility and results in shorter distance path. In this study, the visibility function defined by the proposed anomaly score function indicates the degree of the anomaly of the network traffic. More anomalous traffic results in high anomaly score and then higher pheromone.

The intensity of the pheromone is updated after each cycle of path exploration. A portion of the current pheromone will be evaporated and more pheromone will be accumulated, if the path is explored in the next time frame. The following formula expressed the above update principle.

$$\tau_{ab}(t+1)=\rho\cdot\tau_{ab}(t)+\Delta\tau_{ab}(t,t+1),$$

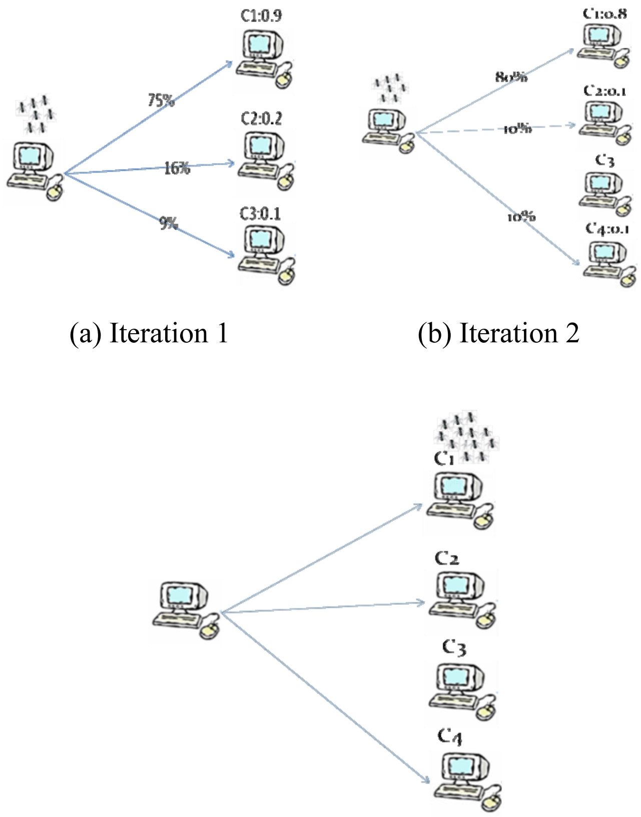
where $\tau_{ab}(t)$ is the intensity of the pheromone on the path *(Sip$_a$, Dip$_b$)* at time frame *t*, $\Delta\tau_{ab}(t,t+1)$ is the accumulated pheromone by the ants during the time frame t to t+1, ρ represents the evaporation rate of the pheromone. The accumulated pheromone is to sum up the pheromone laid by the ants exploring the path during the next time frame. $\Delta\tau_{ab}(t,t+1)$ can be obtained by the following expression.

$$\Delta\tau_{ab}(t,t+1)=\sum_{e=1}^{m}\Delta\tau_{ab}^{e}(t,t+1),$$

where the quantity of the pheromone laid by the e-th ant is defined as $\Delta\tau_{ab}^{e}(t,t+1)=F_{ab}$. Each time when all ants complete one iteration (cycle), the intensity of the pheromone on each path will be recalculated based on the above equations. The proposed detection scheme iterates until the tour counter reaches the pre-defined number of cycles. Once the traffic of all client machines in the network have been explored by ants, the amount of the pheromone collected by each destination from multiple sources is summed up, which represents the anomaly degree of the destination. The suspicious botnet servers are the ones with high pheromone, while normal ones have lower.

To illustrate the proposed ACO-based detection algorithm, Fig. 3 demonstrates how ants explore the paths. Let C1 be a C&C server and C2 to C4 be normal servers. The anomaly scores of the client to the servers are shown beside the server ID. For example, in iteration 1 shown in Fig. 3(a), the anomaly scores of the client to the servers C1 to C3 are 0.9, 0.2, and 01, respectively. The higher value of anomaly score indicates that the connection to C1 exhibits more botnet behavior. Based on path exploration, the probabilities to explore the three paths are 75%, 10%, and 9%, respectively. In the next time frame, shown in Fig. 3(b), where the ACO algorithm proceeds to the next iteration, iteration 2, the client connects to three servers and the connection to C1 still shows more anomaly than others. The ACO computes the probabilities of path exploration: 80%, 10%, and 10% to C1, C2, and C4. After a number of iterations, shown in Fig. 3(c), the algorithm converges and all or most of the ants reach to C1, the C&C server.

After ants finish exploring all the nodes in the network, the explored paths are ordered by the density of pheromone and the destinations with high pheromone are suspected C&C servers. The complexity analysis of ACO algorithm can be found in [24]. The complexity analysis of the proposed ACO-based detection is similar to the above one.



(a) Iteration 1 (b) Iteration 2

(c) Last Iteration

Fig. 3. Illustration of detection

4 Performance Evaluation

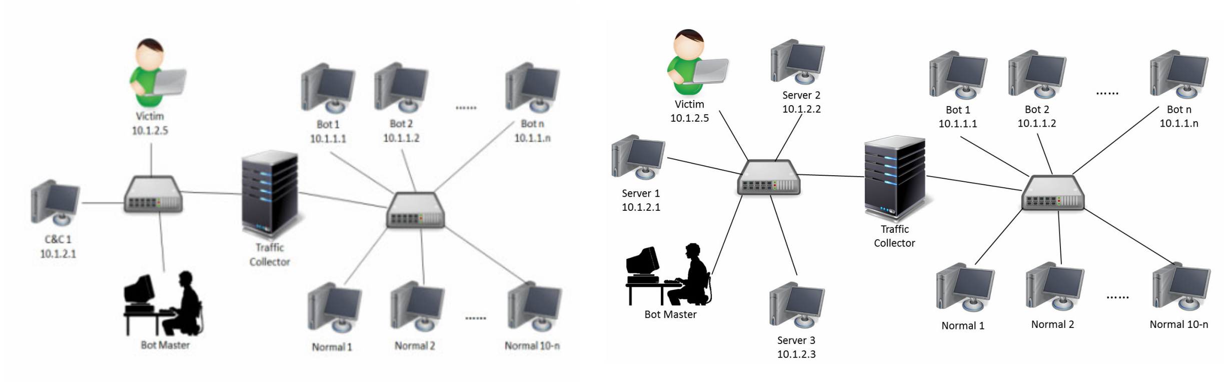
To evaluation the detection performance of the proposed detection system under various network environments, simulated network environments were conducted on a testbed, Testbed@TWISC [25]. The simulated networks consist of the following machines: a botmaster, a victim, a traffic collector, a number of C&C servers, and multiple bots and normal machines.

4.1 Experiment I

Exp I is to evaluate the detection performance of the proposed method under various infection rates in a network. Exp I is further divided into two parts: Exp I.1 has only one C&C server (10.2.1.1) and Exp I.2 contains 3 C&C servers (10.1.2.1~10.1.2.3), where the simulation network environments are illustrated in Fig. 4. To observe if the proposed detection algorithm can identify the C&C servers efficiently, all the experiments were blended in various amounts of malicious and normal (including peer-to-peer) traffic.

To observe if the proposed ACO-based detection system can identify bots and the C&C servers in a network with very few number of infections and little amount of malicious traffic. Therefore, Exp I evaluates if the proposed system can identify the C&C server where the botnet has one C&C server in the network under different infection rates.

The detection results were summarized in Table 2. The results show that the proposed detection method could discover the command and control servers correctly even with a small amount of anomaly traffic.



(a) Exp I.1

(b) Exp I.2

Fig. 4. Network topology of Exp I

Table 2. Detection performance of Exp I

	Bots:Normal	Traffic ratio of M:N:P	No. of malicious	No. of suspicious (detected)
Exp I.1.a	1:9	3:97:0	1	1
Exp I.1.b	5:5	3:97:0	1	1
Exp I.1.c	10:0	3:7:0	1	1
Exp I.2.a	3:7	1:10:0	3	3
Exp I.2.b	3:13	3:75:2	3	3
Exp I.2.c	10:0	3:7:0	3	3

4.2 Experiment II

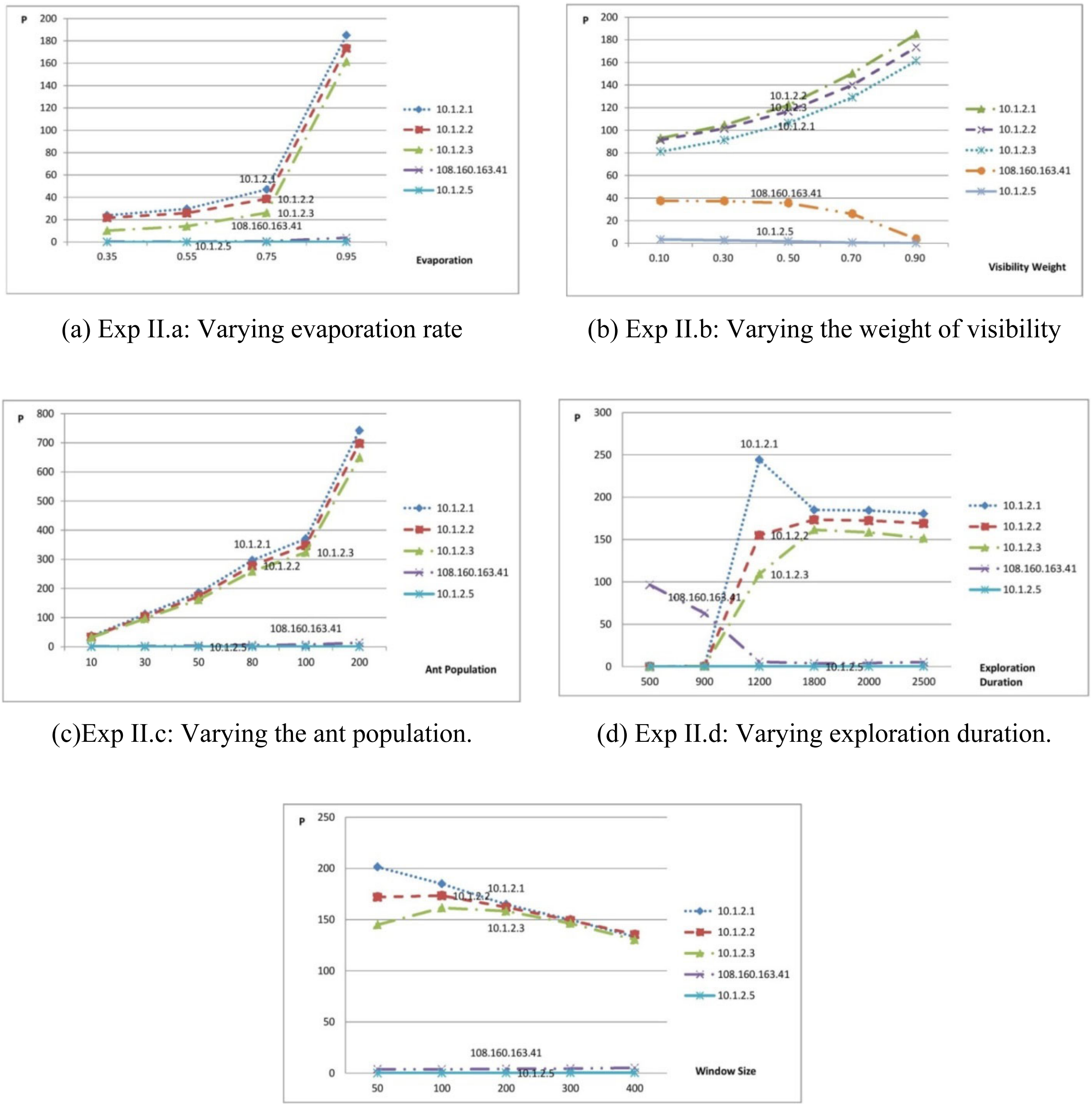
The purpose of Experiment II is to evaluate the sensitivity of the parameters adopted in the proposed detection algorithm. In order to observe the pheromone change in different types of traffic (bot, normal, or peer-to-peer), the simulated network contains three bot machines (10.1.2.1~10.1.2.3), one normal machine (10.1.2.5), and one peer-to-peer node (108.160.163.41) and the traffic ratio (M:N:P) is about 1:4:1. Table 3 summarizes the parameter settings of the experiment Exp II and the results are illustrated in Fig. 5.

Exp II.a varies evaporation rate. Pheromone is long-term memory of an ant colony. Small evaporation rate means low evaporation and slow adaptation. To identify malicious servers efficiently with small amount of malicious traffic, the proposed solution chooses fast adaptation which can accumulate more pheromone on suspicious servers. The results shown in Fig. 5(a) indicate that the proposed detection method can distinguish the malicious servers better when the evaporation rate gets larger.

Exp II.b observes the impact of visibility weight β. According to the results shown in Fig. 5(b), the discrepancy between C&C servers and normal servers increases when the visibility weight becomes heavy. Heavy weight means the anomalous traffic observed in a short time frame affects heavily on the value of pheromone. The discrepancy is still distinguishable even when the visibility weight is small. As the traffic anomaly continues for a certain period of time, the pheromone would lay more on the servers exhibiting anomalous traffic even when the visibility weight is small. The results conclude that the proposed system performs steady and efficiently and is insensitive to the visibility weight.

Table 3. Parameter settings of Exp II

	Evaporation rate	Visibility weight β	Ant population	Exploration duration (sec)	Win size (sec)
Exp II.a	varying	0.9	50	1800	100
Exp II.b	0.95	varying	50	1800	100
Exp II.c	0.95	0.9	varying	1800	100
Exp II.d	0.95	0.9	50	varying	100
Exp II.e	0.95	0.9	50	1800	varying



(a) Exp II.a: Varying evaporation rate

(b) Exp II.b: Varying the weight of visibility

(c)Exp II.c: Varying the ant population.

(d) Exp II.d: Varying exploration duration.

(e) Exp II.e: Varying window size

Fig. 5. The results of evaluating different parameters

Exp II.c evaluates the sensitivity of ant population. The proposed ACO-based detection algorithm accumulates more pheromone if more ants are applied in each pass, as more ants contribute more pheromone along the paths. The experimental results shown in Fig. 5(c) also demonstrate the common phenomenon exhibited on ant colony optimization algorithms; the proposed detection does not need many ants; 50 ants are enough to show the pheromone discrepancy among malicious and normal traffic.

Exp II.d observes the change of exploration duration. Large amount of traffic connections in a short period of time might result in increasing pheromone in case of small exploration duration, while large exploration duration could observe averaged traffic connections. Therefore, the results shown in Fig. 5(d) indicate that the instability of pheromone when the exploration duration is small, less than 1200 and the pheromone discrepancy among different types of traffic becomes stable when the exploration duration gets larger, say larger than 1800. The exploration duration values in between exhibit more discrepancies as short duration accumulates high pheromone.

Exp II.e evaluates the change of the window size which is used for anomaly score computation. The results shown in Fig. 5(e) demonstrate that the window size affects the detection performance insignificantly. The pheromone slightly decreases when the window size gets larger, but it is still distinguishable between the malicious and normal.

The results of two sets of experiments conclude that the proposed detection method could detect malicious C&C servers efficiently under various infection rates.

4.3 Performance Comparison

The botnet detection method proposed by Takemori et al. [26] relied on the victims found by anti-virus software. The functional comparison with Takemori's is summarized in Table 4. Huang [27] proposed a web botnet detection method and could identify fast-flux botnets. The performance comparison is summarized in Table 5. The proposed ACO-based botnet detection outperforms Hunag's on various network environments.

Table 4. Performance Comparison With Takemori'S

	Ours	Takemori's
Rely on info from other hosts	X	O
Keyword comparison	O	O
Packet similarity	O	X
Packet regularity	O	X

Table 5. Performance comparison with Huang's

	Bots:Normal	Traffic ratio of M:N:P	No. of malicious	No. of detected (Ours/Huang's)
Exp I.1.a	1:9	3:97:0	1	1/-
Exp I.1.b	5:5	3:97:0	1	1/-
Exp I.1.c	10:0	3:7:0	1	1/1
Exp I.2.a	3:7	1:10:0	3	3/-
Exp I.2.b	3:13	3:75:2	3	3/-
Exp I.2.c	10:0	3:7:0	3	3/3

5 Conclusion

This study develops a novel visibility function of the ant colony optimization algorithm based on the traffic anomaly; therefore, the paths to malicious servers receive high pheromones. The proposed ACO-based detection system requires no priori information of the whole network topology or the flow information of other routers of the whole network and could identify malicious C&C servers in the early stage of botnet infection with a small amount of malicious traffic. It can be deployed in a local area network or cooperate network; it can help the network administrator identify the suspicious botnet servers and the infected machines in the administrated network domain.

The proposed solution was evaluated on simulated network environments with different infection rates. More evaluations can be done using real botnet traffic collected from a large real network. Further investigation can be done by extending to peer-to-peer botnets.

Acknowledgements

This work was supported in part by a research grant from TWISC and MOST under the Grant MOST-106-2221-E-110-017-MY3, MOST-107-2218-E-006-036 and MOST-106-3114-E-261-001.

References

[1] Taiwan Academic Network Computer Emergency Response Team (TACERT) case studies. <http://tacert.tanet.edu.tw/prog/Document.php>, 2016 (accessed 18.08.15).

[2] K. Thomas, Nine bad botnets and the damage they did. We Live Security. <http://www.welivesecurity.com/2015/02/25/

ninebadbotnetsdamage/>, 2015 (accessed 18.08.15).

[3] Trend Micro, Botnet threats and solutions: phishing. <http://anti-phishing.org/sponsors_technical_papers/trendMicro_Phishing.pdf>, 2006 (accessed 18.08.15).

[4] D.K. McGrath, A.J. Kalafut, M. Gupta, Phishing infrastructure fluxes all the way, IEEE Security and Privacy Magazine 7(5)(2009) 21-28.

[5] J. Wu, L. Zhang, J. Liang, S. Qu, Z. Ni, A comparative study for fast-flux service networks detection, in: Proc. Sixth International Conference on Networked Computing and Advanced Information Management, 2010.

[6] C.Y. Huang, Effective bot host detection based on network failure models, Computer Networks 5(2)(2013) 514-525.

[7] H. Choi, H. Lee, H. Kim, BotGAD: detecting botnets by capturing group activities in network traffic, in: Proc. the Fourth International ICST Conference on Communication System Software and Middleware, 2009.

[8] A. Lakhina, M. Crovella, C. Diot, Mining anomalies using traffic feature distributions, in: Proc. ACM SIGCOMM, 2005.

[9] B. AsSadhan, J.M.F. Moura, D.E. Lapsley, Periodic behavior in botnet command and control channels traffic, in: Proc. IEEE Global Communications Conference (GLOBECOM), 2009.

[10] T.F. Yen, M.K. Reiter, Traffic aggregation for malware detection, in: D. Zamboni (Ed.), Detection of Intrusions and Malware, and Vulnerability Assessment (Lecture Notes in Computer Science, vol. 5137), Springer, Berlin, Heidelberg, 2008, pp. 207-227.

[11] M. Akiyama, T. Kawamoto, M. Shimamura, T. Yokoyama, Y. Kadobayashi, S. Yamaguchi, A proposal of metrics for botnet detection based on its cooperative behavior, in: Proc. International Symposium on Applications and the Internet Workshops (SAINT), 2007.

[12] C. Livadas, R. Walsh, D.E. Lapsley, W.T. Strayer, Using machine learning techniques to identify botnet traffic, in: Proc. 31st IEEE Conference on Local Computer Networks, 2006.

[13] W.T. Strayer, R. Walsh, C. Livadas, D.E. Lapsley, Detecting botnets with tight command and control, in: Proc. 31st IEEE Conference on Local Computer Networks, 2006.

[14] S. Kondo, N. Sato, Botnet traffic detection techniques by C&C session classification using SVM, in: Proc. Security 2nd international conference on Advances in information and computer security (IWSEC), 2007.

[15] W. Lu, G. Rammidi, A.A. Ghorbani, Clustering botnet communication traffic based on n-gram feature selection, in: Proc. Computer Communications, 2011.

[16] C.Y. Huang, Effective bot host detection based on network failure models, Computer Networks 57(2)(2013) 514-525.

[17] G.H. Lai, C.M. Chen, B.C. Jeng, W. Chao, Ant-based IP traceback, Expert Systems with Applications 34(4)(2008) 3071-3080.

[18] P. Wang, H.T. Lin, T.S. Wang, A revised ant colony optimization scheme for discovering attack paths of botnet, in: Proc. the IEEE International Conference on Parallel and Distributed Systems, 2011.

[19] H.H. Chen, S.K. Huang, LDDoS attack detection by using ant colony optimization algorithms, Journal of Information Science and Engineering 32(4)(2016) 995-1020.

[20] A. Castiglione, R.D. Prisco, A.D. Santis, U. Fiore, F. Palmieri, A botnet-based command and control approach relying on swarm intelligence, Journal of Network and Computer Applications 38(2014) 22-33.

[21] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems 26(1)(1996) 1-13.

[22] G. Upton, An ant colony optimization algorithm for the stable roommates, [senior thesis] Richmond, IN: Earlham College,

2002.

[23] D. Subramanian, P. Druschel, J. Chen, Ants and reinforcement learning: a case study in routing in dynamic networks, in: Proc. International Joint Conference on Artificial Intelligence, 1997.

[24] F. Neumann, D. Sudholt, C. Witt, Computational complexity of ant colony optimization and its hybridization. <http://ls2-www.cs.tu-dortmund.de/~sudholt/chapterACO09.pdf>, 2009 (accessed 18.08.15).

[25] Testbed@TWISC, A hybrid network testbed. < http://testbed.ncku.edu.tw/>, 2008 (accessed 18.08.15).

[26] K. Takemori, M. Fujinaga, T. Sayama, Host-based traceback; tracking bot and C&C server, in: Proc. the Third International Conference on Ubiquitous Information Management and Communication, 2009.

[27] M. Huang, Hybrid Botnet Detection, [master's thesis] Kaohsiung, Taiwan: National Sun Yat-sen University, 2010.