A Network Behavior-Based Botnet Detection Mechanism Using PSO and K-means

SHING-HAN LI, Department of Accounting Information, National Taipei University of Business
YU-CHENG KAO, ZONG-CYUAN ZHANG, and YING-PING CHUANG,
Department of Information Management, Tatung University
DAVID C. YEN, School of Economics and Business, SUNY College at Oneonta, Department of Information Systems and Analytics

In today's world, Botnet has become one of the greatest threats to network security. Network attackers, or Botmasters, use Botnet to launch the Distributed Denial of Service (DDoS) to paralyze large-scale websites or steal confidential data from infected computers. They also employ "phishing" attacks to steal sensitive information (such as users' accounts and passwords), send bulk email advertising, and/or conduct click fraud. Even though detection technology has been much improved and some solutions to Internet security have been proposed and improved, the threat of Botnet still exists. Most of the past studies dealing with this issue used either packet contents or traffic flow characteristics to identify the invasion of Botnet. However, there still exist many problems in the areas of packet encryption and data privacy, simply because Botnet can easily change the packet contents and flow characteristics to circumvent the Intrusion Detection System (IDS). This study combines Particle Swarm Optimization (PSO) and K-means algorithms to provide a solution to remedy those problems and develop, step by step, a mechanism for Botnet detection. First, three important network behaviors are identified: long active communication behavior (ActBehavior), connection failure behavior (FailBehavior), and network scanning behavior (ScanBehavior). These behaviors are defined according to the relevant prior studies and used to analyze the communication activities among the infected computers. Second, the features of network behaviors are extracted from the flow traces in the network layer and transport layer of the network equipment. Third, PSO and K-means techniques are used to uncover the host members of Botnet in the organizational network. This study mainly utilizes the flow traces of a campus network as an experiment. The experimental findings show that this proposed approach can be employed to detect the suspicious Botnet members earlier than the detection application systems. In addition, this proposed approach is easy to implement and can be further used and extended in the campus dormitory network, home networks, and the mobile 3G network.

Categories and Subject Descriptors: K. [**Computing Milieux**]; K.6 [**Management of Computing and Information Systems**]; K.6.5 [**Security and Protection (D.4.6, K.4.2)**]

General Terms: Security, Algorithms

Additional Key Words and Phrases: Botnet, particle swarm optimization, K-means clustering, network traffic analysis

---

This work is partially supported by the Ministry of Science and Technology, Executive Yuan, Taiwan R.O.C. (MOST 102–2410-H-015-MY2).
Authors' addresses: S.-H. Li, Department of Accounting Information, National Taipei University of Business, 321 Jinan Road, Section 1, Taipei 104, Taiwan; email: shli@ntub.edu.tw; Y.-C. Kao, Z.-C. Zhang, and Y.-P. Chuang, Department of Information Management, Tatung University, 40 ChungShan North Road, 3rd Section, Taipei 104, Taiwan; emails: ykao@ttu.edu.tw, {raymond0820, eapprendre}@gmail.com; D. C. Yen, School of Economics and Business, SUNY College at Oneonta, Department of Information Systems and Analytics, Oneonta, NY 13820; email: David.Yen@oneonta.edu.

DOI: http://dx.doi.org/10.1145/2676869

ACM Reference Format:
Shing-Han Li, Yu-Cheng Kao, Zong-Cyuan Zhang, Ying-Ping Chuang, and David C. Yen. 2015. A network behavior-based botnet detection mechanism using PSO and K-means. ACM Trans. Manag. Inform. Syst. 6, 1, Article 3 (February 2015), 30 pages.
DOI: http://dx.doi.org/10.1145/2676869

1. INTRODUCTION

Information security issues such as Distributed Denial of Service (DDoS) [Turner 2014], spam, computer viruses and worms, malware, and phishing attacks take place and can be observed often in cyberspace. The German Honeynet Project [2005] stated that among the various forms of malicious software, Botnets in particular have been distinguished as one of the premier threats to computing assets [Gu et al. 2007]. Using the thousands of computers illegally controlled by massive Botnets, attackers paralyze the basic network infrastructures in advanced countries by launching DDoS attacks [Carr 2009]. To this end, the increasing Botnet-infected hosts no doubt introduce an amazing destructive force into the network. It is widely believed that security and convenience are two sides of the same coin, and therefore enhanced security may cause increased inconvenience for users. However, the findings of Kim and Park [2012] determined that advanced technology sometimes achieves enhanced security without much reduction in convenience for some information-related products, such as software and IT services. Further, in addition to the increasing convenience in accessing information, Internet service providers face more challenges than before to improve their service quality in general and security protection in particular.

Botnet has been one of the greatest threats to Internet security [Al-Hammadi et al. 2008; Huang et al. 2011; Tiirmaa-Klaar et al. 2013], and consequently it has long become a topic of public scrutiny. In the past, serious Botnet attacks have occurred. For example, in 2009, a Botnet attack utilized a huge number of infected computers to launch DDoS attacks to penetrate and disrupt the network services of well-known websites, including Twitter, Facebook, Live Journal, and Amazon [Acohido 2009; Whitney 2009]. In 2010, a security firm named NetWitness discovered a new type of malware called Kneber, which had infected more than 75,000 computer systems located/installed in more than 2,500 organizations across the world [NetWitness 2010]. Another case in 2011 found that the crime group named Rove Digital had used Botnet to illegally gain up to 14 million U.S. dollars, and there were more than 4 million computers infected in over 100 countries worldwide [Hacquebord 2011]. In 2013, more than 2 million passwords from social networking sites including Facebook, Google, and Yahoo were stolen by using a Botnet called “Pony,” and such similar attacks would still be continued in 2014 [McAfee 2014]. In addition, with the widespread use of mobile devices, mobile Botnet [Karim et al. 2014] has gradually become a major security issue for cellular networks [Geng et al. 2012].

Obviously, even with the various state-of-the-art malware and virus detection technologies, the Botnet attacking is still regarded as a big threat in the Internet worldwide. According to the Botnet infection statistics by Symantec (as shown in Table I), in the years of 2012 and 2013, the United States was one of the most Botnet-infected countries in the world [Symantec 2014]. Moreover, just in 2013, approximately 76% of spam emails were distributed by spam-sending Botnets [Symantec 2014], and around 19% of mobiles were in fact infected with the Botnet virus through the connection to some unknown websites [F-Secure Labs 2014]. This aforementioned fact shows that the threat of Botnet will have continuous accelerating growth. From this discussion, enhancing Internet security and finding a good solution to handling the Botnet attack becomes a very important task.

Table I. Botnet Infection Statics of Symantec (2012–2013) [Symantec 2014]

Country/Region	2013 Bots Rank	2013 Bots Percentage	2012 Bots Rank	2012 Bots Percentage
United States	1	20.0%	1	15.3%
China	2	9.1%	2	15.0%
Italy	3	6.0%	5	7.6%
Taiwan	4	6.0%	3	7.9%
Brazil	5	5.7%	4	7.8%
Japan	6	4.3%	6	4.6%
Hungary	7	4.2%	8	4.2%
Germany	8	4.2%	9	4.0%
Spain	9	3.9%	10	3.2%
Canada	10	3.9%	11	2.0%

In general, Botnet attacks the communications within a network through the use of destructive malicious software. With the ever-updating and consistent improvement of malicious programs that can be easily planted into the client computers to construct a Botnet, antivirus applications have become more vulnerable and less effective in preventing client hosts from being attacked by Botnet. This study does not focus on developing a detection system like antivirus software, but instead concentrates on uncovering the members of bot clients according to the abnormal network behaviors of each host. So far, most network analyses use packet contents and network flow characteristics to analyze the behavior of data. However, these aforementioned approaches may still have some challenging problems to be resolved. For example, the packet contents and flow characteristics can be easily changed to avoid the detection of/by the system. Further, the malware programs can be easily edited [Masud et al. 2011] or modified to change the flow characteristics through the utilization of different sizes of packets and/or the adjustment of packet transmission speed per second during communication. Once the packet contents are changed by adding extra, useless characters, the method based on packet traffic features must be improved further to learn new feature patterns, and, consequently, the difficulty of analyzing and studying the encrypted packets will be doubled. In addition, users have become increasingly more concerned about their privacy and may not allow the packet contents involving a large amount of their personal information to be directly accessed by others [Li 2012].

This study proposes a mechanism based on Particle Swarm Optimization (PSO) and K-means clustering algorithms to resolve these aforementioned problems. This study also justifies that the Botnet-infected computers may contain at least three different behaviors within the network activities, even if the packet contents or the traffic characteristics of transmission have been changed. This mechanism aims to locate and identify the members of the network in an organization and aims to accomplish the following three objectives for Botnet detection. First, this mechanism is developed to uncover the computers infected by the malicious Botnet in the private network of an organization. Second, by adopting this approach, Botnet can still be detected effectively, even though the packet contents or the network traffic characteristics have been encrypted or changed. Third, the mechanism can detect the Botnet-infected computers earlier than the other intrusion detection systems. Specifically, this study conducts an experiment on traffic flows of a campus network and then evaluates the efficiency and the capability of the prediction according to the analysis results. The obtained evaluation results prove that the three aforementioned benefits can be accomplished by adopting this proposed mechanism.

The remaining part of this article is organized as follows. Section 2 reviews the prior literature related to the Botnet detection approaches and available applications of PSO + K-means. A Botnet detection mechanism based on PSO and K-means clustering is proposed and developed subsequently in Section 3. Section 4 uses a case study to implement this mechanism and assesses the results of the analysis. Finally, the last section provides a conclusion with some suggestions for future studies.

2. LITERATURE REVIEW

2.1. Botnet Detection

Botnet detection has been a widely discussed topic in many studies related to network security. The researchers have proposed different strategies to improve the efficiency and effectiveness for Botnet detection. The study of Strayer et al. [2006] used statistical flow characteristics and supervised classifiers to develop a mechanism that studied the flow characteristics to exclude the unlikely traffic flows of Botnet, classify the remaining traffic into a group that is likely to be part of a Botnet, and then correlate the potential traffic to determine the most likely activity of Botnet. The results of this study show that the malicious Botnet can be detected by finding a common IP address endpoint and any possible evidence of communication that exists between the Botmaster and the command and control (C&C) server [Strayer et al. 2006].

The studies of Gu et al. presented three strategies to detect the potential Botnet infections in network traffic and implemented the prototype systems according to three proposed approaches: BotHunter [Gu et al. 2007], BotSniffer [Gu et al. 2008a], and BotMiner [Gu et al. 2008b]. Specifically, BotHunter is designed to track the two-way communication flows between internal assets and external entities and employed to understand the life cycle of malware infections through Intrusion Detection System (IDS)-driven dialog correlation. This system collects the main information about malicious behaviors, such as scanning, exploits usage, egg downloading through a security tool—SNORT, Statistical payLoad Anomaly Detection Engine (SLADE), and Statistical sCanAnomaly Detection Engine (SCADE). The system then analyzes these correlated behaviors to uncover Botnet-infected client hosts. In short, BotHunter implements an analytical model that can correlate multiple behaviors rather than just detect the traffic flows by using a single characteristic.

Further, BotSniffer [Gu et al. 2008a] contains two components: the monitor engine and the correlation engine. This program can be used to examine network traffic, generate a connection record of suspicious C&C protocols, and detect activity response behaviors (e.g., scanning, spamming). For the scan activity detection, BotSniffer uses approaches similar to SCADE, which was developed for BotHunter [Gu et al. 2007]. Specifically, the authors developed port-independent protocol matchers to find all suspicious Internet Relay Chat (IRC) and associated HTTP traffic. BotSniffer basically performs group analysis to capture the information of spatial-temporal correlations in network traffic and utilizes statistical algorithms to detect suspicious Botnets with a high accuracy and also a very positive low-fault rate.

Gu et al. [2008b] improved the previous Botnet detection approaches that worked on specific Botnet C&C protocols (e.g., IRC) and structures (e.g., centralized) to present a new detection framework that is independent of the Botnet C&C protocol and structure, clusters similar communication traffic and similar malicious traffic, and correlates the aforementioned two traffic results to identify the hosts that share both similar communication patterns and similar malicious activity patterns. The results of this study show that BotMiner can detect real-world Botnets (IRC-based, HTTP-based, and P2P Botnets including Nugache and Storm worm) and has a very low false detection rate.

The study of Zeng et al. [2010] presented a framework that combined both network- and host-level information to detect Botnets. This approach can be used to detect Botnets that appear stealthy in network activities with the assistance of host-level information and to extract features from NetFlow data to analyze the similarity or dissimilarity of network behaviors without further inspection of each packet's payload. This detection relies on the invariant properties of Botnets' network and host behaviors, which are independent of the underlying C&C protocol so that it can be employed to detect both traditional IRC and HTTP and hybrid P2P Botnets. In summary, the results of this study show that their proposed framework can be used to detect different types of Botnets with low false-alarm rates.

In addition, the study of Zhang et al. [2010] proposed an approach to Botnet detection based on PSO techniques. For this approach, the first step is to capture the collection of malware data set through the Honeypot system and classify data into different types, such as SDBot, MyBot, PoeBot, IRCBot, and unknown bot. The next step is to define 16 flow characteristics for each flow in the traffic traces and eliminate traffic flows that are unlikely to be part of Botnet. The last step is to perform the data training and data-classification-based PSO mechanism. To explore the effectiveness of PSO-based classification in identifying IRC traffic, this study employs two comparable classification techniques: J48 (i.e., the WEKA implementation of C4.5 decision trees) and Bayesian networks. The experimental results show that it reaches a higher accuracy with the PSO mechanism.

The study of Lu et al. [2011] proposed a new approach for detecting and clustering Botnet traffic flows on large-scale network application communities. This approach is a hybrid mechanism that conducts two-stage classifications. To start with an introduction, a signatures-based classifier is used to classify the network traffic into different applications. The packet payload is then classified into 256 dimensions representing the different flow features. The unknown flows that cannot be identified by the signatures-based classifier are classified again by applying them with a novel decision tree model. The network traffic is clustered based on n-gram features extracted from the content of network flows to differentiate the malicious Botnet traffic created by bots from normal traffic on each specific application. In summary, this proposed approach can detect two IRC Botnet traffic traces with a high detection rate and ensure an acceptable low false-alarm rate.

The study of Zhao et al. [2013] proposed an efficient approach to detect Botnet by extracting from the datasets of network flow with 12 features including an average payload packet length, the size of the first packet in the flow, and so forth. In addition, this approach utilizes a machine-learning method to analyze the detected results. Because of its utilizing the features of network flow instead of analyzing the payload packets directly, this study provides a much higher security alternative to uncover all types of notable and unnotable Botnet activities.

In summary, the study of Gu et al. [2008b] mainly uses the records of the IDS and analyzes the decrypted packet contents to determine the normal and malicious behaviors. However, there exist three main problems with the study's approach for the analysis of packet contents. First of all, packet contents are actually involved with security and privacy issues [Zeng et al. 2010]; thus, if the contents are directly used for analysis, the costs of calculation will certainly increase. Second, it proves to be a big challenge to analyze the encrypted packets of Botnet. Third, packet features can be easily changed by just adding some useless characters to the contents, and therefore, the suspicious hosts cannot be accurately detected by simply analyzing the packet contents. Another approach proposed later by Zeng et al. [2010] can only be used in the host environment where the detection programs have been installed. In other words, if the computers are unable to install the sensors, this method will turn out to be unavailable. On the other hand, the study of Strayer et al. [2006] focuses on the IRC-based Botnet, and it is rather limited, since it may not be applicable to other types of Botnet. Further, Zhang et al. [2010] apply a machine-learning algorithm approach

Table II. Comparisons Between Prior Studies and This Study

Related Works	Host Data	Packet Contents in Application Layer	Data in Network and Transport Layers
This proposed mechanism			Y
Strayer et al. [2006] Detecting Botnets with tight command and control			Y
Gu et al. [2007] BotHunter: detecting malware infection through IDS-driven dialog correlation		Y	
Gu et al. [2008] BotSniffer: Detecting Botnet command and control channels in network traffic		Y	Y
Gu et al. [2008] BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection		Y	Y
Zeng et al. [2010] Detection of botnets using combined host- and network-level information	Y		Y
Zhang et al. [2010] Exploring a swarm intelligence methodology to identify command and control flow		Y	
Lu et al. [2011] Clustering botnet communication traffic based on n-gram feature selection		Y	
Zhao et al. [2013] Botnet detection based on the traffic behavior analysis and flow intervals			Y
The proposed approach presented in this study			Y

for clustering and classifying the traffic flows, and the current Botnet would then be ignored if no new patterns could be located.

Comparing with the approach proposed by the study from Zhao et al. [2013], this study extracted the important symptoms using network behaviors rather than the network flow data. Consequently, this proposed approach can detect the Botnets more successfully even if some data features/symptoms are changed by Botnet malware to prevent the detection systems from locating those malicious activities.

Table II shows the differences between these aforementioned studies and the current article. Specifically, it includes such attributes as host data, packet contents in the application layer, and data in the network and transport layers.

- "Host data" are the information or records gathered by the program installed in the client computer.
- "Packet contents in the application layer" refers to the flow characteristics extracted from the packet payload, which are the data employed for analysis.
- "Data in the network and transport layers" contain network layer IP or Port used in both TCP and UDP traffic flows in the transport layer.

2.2. Applications of PSO + K-means

PSO is a new evolutionary computation technique originally introduced by Kennedy and Eberhart [1995] and Eberhart and Kennedy [1995]. It is regarded as an efficient heuristic aimed at solving hard optimization problems [van den Bergh and Engelbrecht 2006]. The development of this approach is mainly based on the simulation of social behavior of animals or humans, including a flock of birds, a school of fish, and a group of people who may pursue a common goal in their lives. In addition, the PSO algorithm may be somewhat similar to the genetic algorithm (GA). Further, the PSO algorithm may be utilized to enhance the population evolution as a substitute of GA. For example, the problem of migration among subspecies of robots can be a challenge to cause GA crossover, but it should not exist with particle swarms [Eberhart and Kennedy 1995].

PSO is, in fact, a random-search approach based on swarm intelligence [Kennedy and Eberhart 1995, 2001], and its search process can be described as particles being "flown" through a hyperdimensional search space by means of adjustment of their positions in the search space [Kennedy et al. 2001; Messerschmidt and Engelbrecht 2004]. Each particle in the swarm actually represents a candidate solution to the optimization problem [Zhan et al. 2009] and uses an objective function to calculate the fitness value to evaluate the quality of a solution.

Each particle in PSO uses the objective function to determine its fitness value and then calculates iteratively to find out the optimal solution through a cognitive-only model or social-only model [Kennedy 1997]. The former one determines the best optimal value of (*pbest*) between the current fitness value of particles and also the best value of particles from the past searching process. The (*pbest*) is then recorded in the memory of each particle and it may adjust its search direction accordingly based on the current pbest. As to later one, the (*gbest*) is searched out from the search process of PSO particles, and then the better one between (*pbest*) and (*gbest*) for the best group value is determined. Generally speaking, the social-only model is faster and more efficient than the cognitive-only model [Kennedy 1997].

Compared to the traditional random search algorithms, PSO is a strategically random rather than a completely random search mechanism. Due to the nature of fast convergence, the PSO algorithm can quickly find out the most optimal value in the solution space and resolve the NP-complete problems during a reasonable time period. In addition, the PSO algorithm may be applied to many subject areas such as resolving graphic recognition, clustering optimization, scheduling assignments, conducting linear programming, and performing network optimization and multiobjective optimization.

On the other hand, K-means [Hartigan and Wong 1979] is one of the most popular and simple clustering algorithms, and it was first introduced in 1955. The data-clustering technique mainly puts the data with similar attributes into the same cluster and therefore the data grouped in different clusters may have totally different characteristics. In the field of data mining, this aforementioned method is also referred to "unsupervised learning." Comparatively, a "supervised learning" method such as "classification" needs predefined types and also training data for each type, and then data is classified into the same group according to the attributes of those defined types [Han 2001]. To this end, the data-clustering technique is widely applied in the fields from business to biology. Furthermore, it can be used in performing the anomaly analysis to find out the outlier value and help the auditor to effectively trace the suspicious data.

The K-means algorithm may be deemed as the most commonly used partition clustering algorithm simply because it is easily implemented and is by nature the most efficient clustering method in terms of execution time [Cui and Potok 2005]. Over a time span of 50 years, K-means has been and is still widely used [Jain 2010] and is mainly applied to exploit the K centroids for clustering problem [Lam et al. 2012]. Specifically, this algorithm may include the following realization steps [Lin et al. 2012]: (1) to arbitrarily select K data points from the dataset as initial cluster centers, (2) to calculate the distance from every data point to each of the K centers and classify them into the nearest cluster, (3) to recalculate the mean value of each cluster as the cluster center, and (4) to end the algorithm if the division results no longer change or reach the maximum number of iterations, or return to step 2 otherwise.

The PSO-based K-means clustering algorithm (known as PSO-KM) is becoming a popular basic strategy to be applied in order to tackle the clustering problem. This algorithm has a good overall convergence and can effectively overcome the shortcoming; the traditional K-means clustering algorithm is easy to be converged at the local minima [Li et al. 2011]. The studies of Du et al. [2008] and Sun et al. [2012] introduced an improved variation to this aforementioned basic strategy to achieve a better performance [Lam et al. 2012].

There are many studies using this hybrid clustering method in different fields. Rana et al. [2010] proposed a new hybrid sequential clustering approach based on PSO and K-means that overcomes the drawbacks of both algorithms. Cui and Potok [2005] combined PSO with K-means to present a document-clustering mechanism that performs fast document clustering and avoids being trapped in a local optimal solution as well. This hybrid algorithm combines the capability of globalized searching of the PSO algorithm and the fast convergence characteristic of the K-means algorithm. Ahmadyfard and Modares [2008] presented that the combined method (PSO-KM) does have the advantage of both PSO and K-means methods. In their study, the result of an experiment on five datasets including real and synthetic data proved that the hybrid algorithm outperforms K-means and PSO clustering to enhance the resulting data clustering. Li et al. [2011] proposed an anomaly intrusion detection method based on the PSO+K-means algorithm and used datasets KDD CUP 99 to conduct further experiments. The result shows that the proposed method has a higher detection rate and a lower false detection rate. Benson and Shalini [2012] proposed an architecture that depends on the PSO-based IP backtracking. The experimental results from the aforementioned studies showed that this approach is pretty successful in terms of the prompt detection of Botnets. In addition, the PSO schema with K-means algorithm can be used in a face recognition system that can be accurately employed to identify faces in real time, recognize effectively whether a person's profile is preserved in the database, and detect/recognize each person's identity when an image contains several people [Pai et al. 2012].

Generally speaking, the PSO-KM algorithm obviously improves the efficiency of clustering. From the previous discussion, this study classifies three main network behaviors and extracts the suspicious Botnet clients from normal clients based on this hybrid clustering approach to structure a Botnet detection mechanism, which is described in the next section.

3. BOTNET DETECTION MECHANISM USING PSO+K-MEANS

The architecture of this proposed Botnet detection mechanism including five layers is displayed in Figure 1. First, in the RawData Layer, network traffic entries are collected from various network devices, and the flow records are stored in a centralized database. Second, network flow data are transformed into related information in the DataProcess Layer. Third, audit-purposed IP features are retrieved from the information in the LanIP Layer. Fourth, the similar IP features are categorized into a specific type of network behavior, and by doing so, three types of behaviors can be classified in this mechanism, including ActBehavior, FailBehavior, and ScanBehavior. Last, the abnormal client bots (i.e., Botclient) can be detected in the cluster layer based on the PSO algorithm and the K-means clustering method. To summarize, the details for each aforementioned step of this Botnet detection mechanism are discussed in Sections 3.1 to 3.5.

Table III, which is based on some relevant literature [Gu et al. 2007, 2008; Zhu et al. 2008; Feily et al. 2009; Ollmann 2009; Rodriguez-Gomez et al. 2011; Wang et al. 2011; Rostami et al. 2011], summarizes the Botnet behaviors in different roles (e.g., Botmaster, C&C server, and Botclient) and their related data sources (e.g., data in network

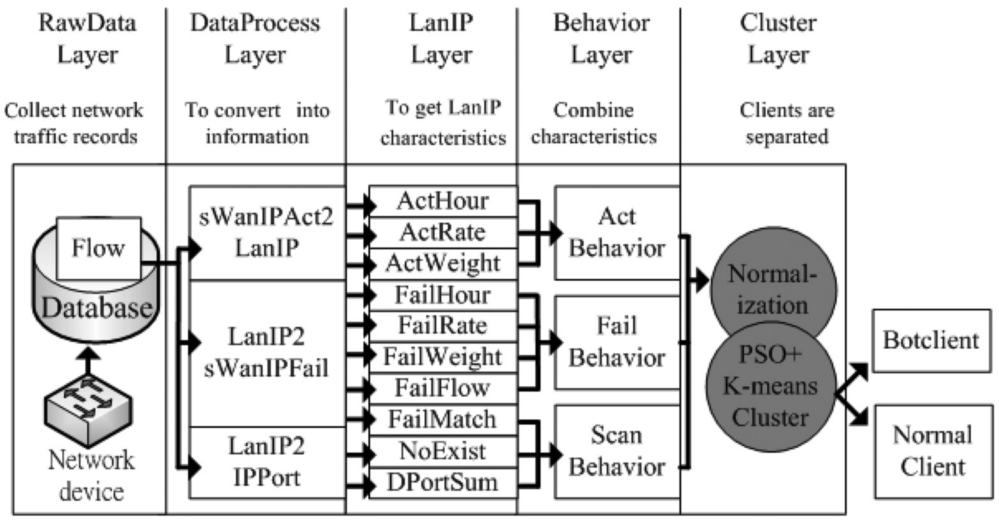

Fig. 1. Architecture of this proposed Botnet detection mechanism.

Table III. Summary of the Botnet Behaviors in Different Roles

		Data Sources		
Role	Behavior	Data on Network Layer	Packet Content in Application Layer	Host Data
Botmaster	Keep communicating with C&C server	Y	Y	
	Make and spread update data		Y	
C&C Server	Keep communicating with Botclient and Botmaster	Y	Y	
	Keep secrecy		Y	
	Keep high availability	Y		
	Get and spread update data		Y	
Botclient	Scan other client	Y	Y	
	Infect other client		Y	
	Get update data		Y	
	Open backdoor in client			Y
	Trigger malware process in client			Y
	Keep communicating with C&C server	Y		
	Change another C&C server	Y		
	DDos or other attack	Y	Y	

and transport layers, packet contents in application layer, and host data). Botclient will return the current status to Botmaster to get the latest instruction. Therefore, Botclient and Botmaster will try to keep connecting with the C&C server. The situation of mutual communication can be observed through analysis of the activities in the network layer. Botnet becomes more robust and more difficult to be detected by updating data files to change the malware signature and listing of the C&C server. However, all updated data are in the packet, so it needs to analyze the packet contents to confirm whether these are the updated files of Botnet. In order to maintain secrecy, Botnet uses DNS to implement a fast flux technique. Because DNS information is included in the packet, it needs to analyze the packet contents to conduct further investigation. In addition, keeping high availability and high secrecy of Botnet would make the C&C

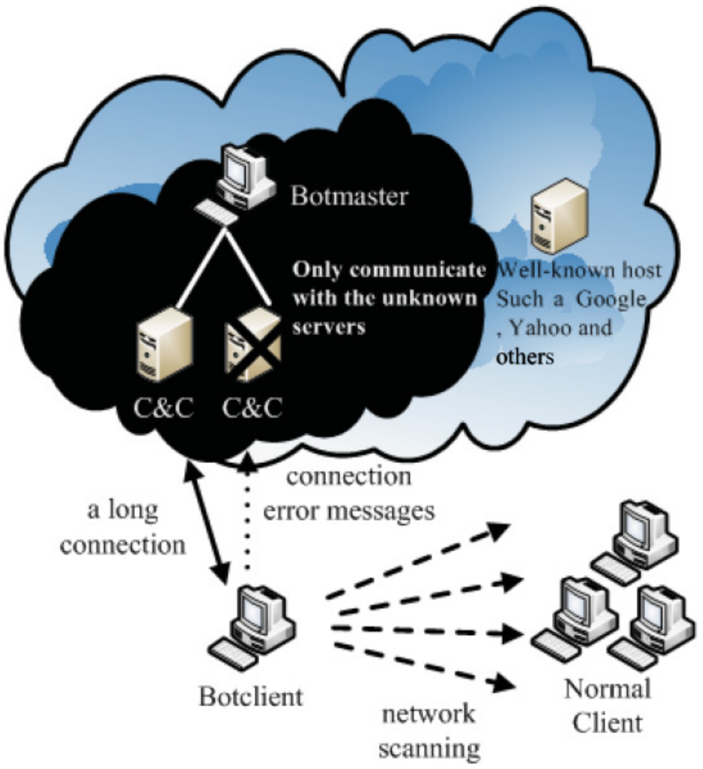
Botmaster
Only communicate with the unknown servers
Well-known host Such a Google, Yahoo and others
C&C
C&C
a long connection
connection error messages
Botclient
network scanning
Normal Client

Fig. 2. Phenomenon of Botclient in the network layer.

server more complicated. On the other hand, when switching to another server, it will connect to a machine that is shut down or on a timeout but not an active C&C server, and plenty of error messages will appear in the network layer. The attacking scope of Botnet will be expanded during infection; therefore, Botclients will scan constantly and attack all hosts in the network. A large number of scans can be noticed through the network layer; however, the invasion behaviors can only be confirmed from the analysis of packet contents. The most common network attack ways, such as DDoS or spam, can be easily detected from the tremendous abnormal traffic flows. As for stealing the confidential information, it might need the further analysis of packet contents to confirm. Furthermore, the length of the incubation period in Botnet is unknown, and not all Botnets have aggressive behaviors; therefore, some Botnets will be sold by a Botmaster to the black market.

Based on the information presented in Table III, Figure 2 is provided to show four major phenomena of a Botclient that occurred in the network layer. Further, the paragraphs that follow provide more discussion about each of these four phenomena:

(1) Long-time communication capability
The Botclient has a critical characteristic of keeping a long and active connection to C&C servers for obtaining a new command and then returning the Botmaster to the current status [Gu et al. 2008].
(2) Continuous prompt delivery of connection error messages
During a situation in which either not every connection is active or the server(s) may be shut down or offline, the Botclient communicates with multiple C&C servers

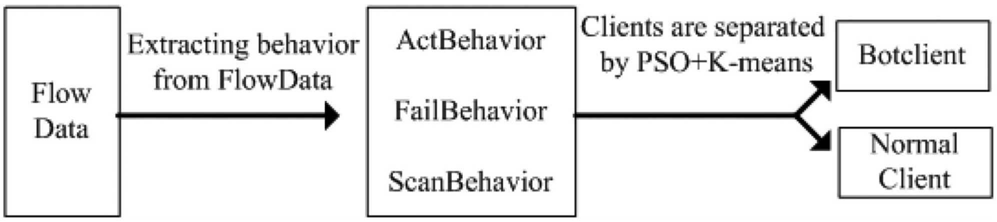
Flow Data
Extracting behavior from FlowData
ActBehavior
FailBehavior
ScanBehavior
Clients are separated by PSO+K-means
Botclient
Normal Client

Fig. 3. Conceptual Botclient detection of this proposed mechanism.

occasionally so that its connection failure messages would continue to occur [Wang et al. 2011; Rostami et al. 2011].

(3) Expand Botnet through large-scale malicious scanning
To expand the Botnet scope, it locates and attacks the vulnerable hosts by scanning several network servers and also installs "backdoors" on them, and eventually those server hosts may turn into members of the Botnet [Gu et al. 2007].

(4) Communicate only with the unknown servers
Generally speaking, the Botmaster only communicates with the unknown servers. It is more difficult to build a C&C server on the computer with a high security control and tight management in general, for such famous companies as Google and Yahoo in particular. On the contrary, the servers with a lower security control are able to be invaded much more easily and consequently turn into a C&C server, such as vulnerable servers, private computers, or unknown server hosts.

Figure 3 can be used to illustrate the conceptual Botclient detection of this proposed mechanism. First, according to the aforementioned four phenomena of Botclients, network traffic flows that only communicate with the unknown servers can be classified into three behaviors: ActBehavior, FailBehavior, and ScanBehavior. Second, suspicious Botclients will be separated from the normal clients by using PSO and the K-means clustering algorithm. The three aforementioned behaviors of Botnet can be defined further as follows:

- ActBehavior: the behavior of an active connection and a long communication with all suspicious servers (i.e., the unknown hosts)
- FailBehavior: the behavior of frequently generating error messages due to the occurring connection failures
- ScanBehavior: the behavior of conducting suspicious scanning

3.1. Collect and Store Network Traffic Records: RawData Layer

In the RawData layer, the traffic flows derived from the network switch are required to be further transferred and stored in the database. The study of Brownlee et al. [1999] used a flow-based approach to collect network flow data, of which the statistical unit is "flow." Moreover, the "flow" is defined as follows:

> *A flow is a stream of packets observed by the meter as passing across a network between two end points, which have been summarized by a traffic meter for analysis purposes*. [Brownlee et al. 1999]

Network flow records are supported by various network devices, such as the NetFlow protocol developed by Cisco Systems, the J-flow protocol of the Juniper network, and the Audit Record Generation and Utilization System (ARGUS) [Piskac et al. 2011]. Flow-based network flows are highly granular. For example, flow records include such details as source IP address, source Port, destination IP, destination Port, packet and byte

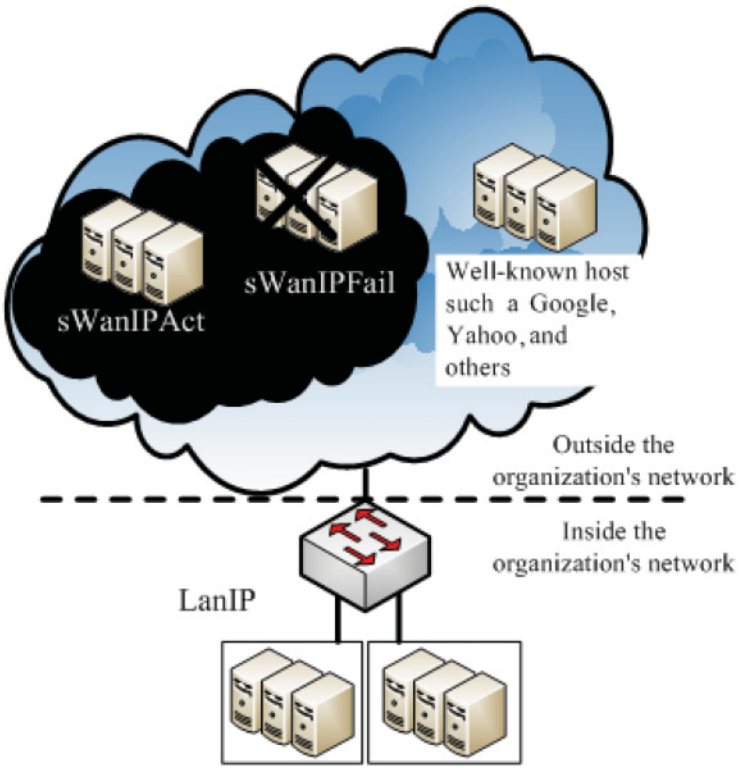
sWanIPAct
sWanIPFail
Well-known host such a Google, Yahoo, and others
Outside the organization's network
Inside the organization's network
LanIP

Fig. 4. IP Locations.

Table IV. Network Flow Fields of This Proposed Mechanism

Field Name	Description
Time	The sending time of the first packet of "flow"
Source IP	The IP of sending messages
Destination IP	The IP of receiving messages
Destination Port	TCP and UDP destination Ports This port number is for destination IP use to receive the messages
Failure signal	Connection failure signal

counts, timestamps, type of service, application ports, and input and output interfaces [Piskac et al. 2011]. Out of these aforementioned details, five attributes, including time, source IP, destination IP, destination Port, and failure signal, are filtered out in this study, which are shown in Table IV.

3.2. Extract the Related Information from Flow Records: DataProcess Layer

The purpose of the DataProcess Layer is to define the IP address parameters, which are shown in Figure 4. The IP to be processed does not refer to a particular IP but to all IP addresses. The first step is to define "LanIP" as the IP addresses to be audited within the organizational network and also refer it as "Lan IP." The second step is to find out all suspicious Wan IP addresses by excluding those of well-known network domains like Google, Yahoo, and so forth, and then determine an IP type according to the connection signal. If this returns a failure signal, the "sWanIPFail" will be defined

Table V. Network Flows of Connection from $sWanIPAct_1$ to $LanIP_1$

Date/Time	Source IP (suspicious Wan IP)	Destination IP (Lan IP)	
2012-04-29 06:00:18	$sWanIPAct_1$	$LanIP_1$	}2 flows in period 6
2012-04-29 06:01:09	$sWanIPAct_1$	$LanIP_1$	
2012-04-29 07:03:03	$sWanIPAct_1$	$LanIP_1$	}3 flows in period 7
2012-04-29 07:25:24	$sWanIPAct_1$	$LanIP_1$	
2012-04-29 07:56:31	$sWanIPAct_1$	$LanIP_1$	
2012-04-29 09:42:41	$sWanIPAct_1$	$LanIP_1$	}1 flow in period 9

Table VI. Network Flows per Hour from $sWanIPAct_1$ to $LanIP_1$

Source IP (Suspicious Active Wan IP)	**Destination IP** (Lan IP)	**Network Flows per Hour (0–23)**															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	22	23
$sWanIPAct_1$	$LanIP_1$							2	3		1						

as an inactive Wan IP. On the other hand, if the connection is successful, "sWanIPAct" is designated for an active Wan IP.

The definitions of IP in the DataProcess Layer are:

- IP: All IP addresses including Lan IP (inside the organization's network) and Wan IP (outside the organization's network)
- LanIP: IP addresses to be audited inside the organization's network, called Lan IP
- sWanIPFail: suspicious inactive Wan IP
- sWanIPAct: suspicious active Wan IP

Furthermore,

$$\text{sWanIPFail,sWanIPAct,LanIP} \in \text{IP};$$
$$\text{sWanIPFail} \cap \text{sWanIPAct} \cap \text{LanIP} = \psi;$$

According to these aforementioned definitions, the related information extracted from daily network traffic flow records can be divided into three parts:

(1) Summarize the daily network flows with which the single suspicious active Wan IP (sWanIPAct) connects to the single Lan IP. Table V shows the following information: Date/Time, Source IP, and Destination IP of the network traffic flows, while Table VI summarizes the data of each hour.

Tables V and VI show all the network flows from a single-source IP to a single-destination IP. In a more practical scenario sense, the connections are actually from a multisource IP to a multidestination IP. Table VII represents the summary of network flows, with which all suspicious active Wan IPs (sWanIPAct) connect to a Lan IP (LanIP).

(2) Summarize the daily network flows with which all Lan IPs (LanIPs) are connected to the single suspicious inactive Wan IP (sWanIPFail). The data process is similar to that of sWanIPAct2LanIP, except that the source IP is a Lan IP and the destination IP is a suspicious inactive Wan IP. The first step will be to collect the information

Table VII. Network Flows from sWanIPAct to LanIP

Source IP (suspicious active Wan IP)	**Destination IP (Lan IP)**	**Network flows per hour (0-23)**															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	22	23
$sWanIPAct_1$	$LanIP_1$							2	3		1						
$sWanIPAct_2$	$LanIP_1$									2	1	8	6				
$sWanIPAct_{j1}$	$LanIP_1$								2	3	4		7	3			
$sWanIPAct_1$	$LanIP_2$					1	8	6									
$sWanIPAct_2$	$LanIP_2$						2	5	3					3	2		
$sWanIPAct_{j2}$	$LanIP_2$						3	1	2							2	
~																	
$sWanIPAct_1$	$LanIP_i$	1	2	1	3	4	1	2	3	2	1	8	6	4	5	2	3
$sWanIPAct_2$	$LanIP_i$											2	5	3			
$sWanIPAct_{ji}$	$LanIP_i$														2		

Table VIII. Records of Connection from $LanIP_1$ to IP_1

Date/Time	Source IP **(Lan IP)**	Destination IP (All IPs)	Destination Port
2012-04-29 06:00:18	$LanIP_1$	IP_1	TCP80
2012-04-29 06:01:09	$LanIP_1$	IP_1	TCP80
2012-04-29 07:03:03	$LanIP_1$	IP_1	TCP443
2012-04-29 07:25:24	$LanIP_1$	IP_1	TCP80
2012-04-29 07:56:31	$LanIP_1$	IP_1	TCP5050
2012-04-29 09:42:41	$LanIP_1$	IP_1	TCP80

Table IX. Used Destination IP of Connection from Private Network IP_1 ($LanIP_1$) to IP_1

Source IP **(Lan IP)**	Destination IP (All IPs)	Used Destination Port
$LanIP_1$	IP_1	TCP80, TCP443, TCP5050

on three key attributes (source IP, destination IP, and destination Port) of network flow records for the connection from a Lan IP (LanIP) to a single suspicious inactive Wan IP (sWanIPFail). Second, the traffic flows of each hour will be summarized. In a more practical scenario, the connections are from multisource IPs to multidestination IPs. Therefore, calculate network flows for each hour, including that all Lan IPs (LanIPs) connect to a single suspicious inactive Wan IP (sWanIPFail), and define it as "LanIP2sWanIPFail."

(3) Summarize the daily network flows with which all Lan IPs (LanIPs) connect to all ever-used destination Ports. The first step is to get all the flow records that the Lan IP (LanIP) connects to a single destination port, as shown in Table VIII. Next, the data is summarized by source IP, destination IP, and distinct destination port value, as Table IX shows.

Tables VIII and IX demonstrate the example record of the communication from a single-source IP and a destination IP. Again, in practical cases, the flow entries

Table X. Used Destination IP of Connection from LanIP to Any IP

Source IP **(Lan IP)**	Destination IP (Any IP)	Used Destination Port
LanIP_1	IP_1	TCP80, TCP443, TCP5050
LanIP_1	IP_2	TCP80
LanIP_1	IP_{j1}	TCP80, UDP4872, …., UDP5683
LanIP_2	IP_1	TCP80, TCP25, …., UDP1813
LanIP_2	IP_2	TCP80, TCP110, …., UDP29910
LanIP_2	IP_{j2}	UDP5001, UDP5002, …., UDP5099
	~	
LanIP_i	IP_1	TCP80, TCP443
LanIP_i	IP_2	TCP3389
LanIP_i	IP_{ji}	TCP80, TCP110, …., UDP18910

consist of the connections from all Lan IPs (LanIPs) to a single destination port that the destination IPs ever used, and "LanIP2IPPort" is designated for describing this information, which is shown in Table X.

3.3. Extract Behavior Features: LanIP Layer

In the LanIP Layer, there are 10 features generated followed by the results of the DataProcess Layer to define three connection parameters, as shown in Table XI. One feature value comes from a multiconnection IP, and the largest value is viewed as the key feature of LanIP. The average value is seldom to be used, simply because it may be reduced and is caused by the smaller feature value.

The formulas of these 10 feature metrics are defined as follows:

(1) **ActHour = {$\mathbf{ActHour_1, ActHour_2, \ldots, ActHour_i}$};**
$\mathbf{ActHour_i} = \mathbf{max}(\boldsymbol{ah_{1i}, ah_{2i}, \ldots, ah_{ji}})$, the $\boldsymbol{i}$th feature value of LanIP;
$\boldsymbol{fh_{i1}}$ = the number of hours that the hourly network traffic flows connect from the jth suspicious active Wan IP (sWanIPAct_j) to the ith Lan IP (LanIP_i), which is greater than 0.

(2) **ActRate = {$\mathbf{ActRate_1, ActRate_2, \ldots, ActRate_i}$};**
$\mathbf{ActRate_i} = \mathbf{max}(\boldsymbol{ar_{1i}, ar_{2i}, \ldots, ar_{ji}})$, the $\boldsymbol{i}$th feature value of LanIP;
$\boldsymbol{ar_{ji}} = \frac{\boldsymbol{ah_{ji}}}{\boldsymbol{((end_i - start_{ji}) + 1) - null_i}}$, the ratio of the number of hours that the jth suspicious active Wan IP ($\text{sWanIPAct}_\text{j}$) connects to the ith Lan IP (LanIP_i), shown in Tables XII and XIII.

- $\boldsymbol{end_i}$ = the last hour of the network activity from all suspicious active Wan IPs (sWanIPAct) to the ith Lan IP (LanIP_i)
- $\boldsymbol{start_{ji}}$ = the first hour of the network activity from the jth suspicious active Wan IP (sWanIPAct_j) to the ith Lan IP (LanIP_i)

Table XI. 10 Features of LanIP

DataProcess Layer	LanIP Layer	Description
sWanIPAct2LanIP	ActHour	Time of communication for the connection from sWanIPAct to LanIP; the higher the value is, the longer the time is.
	ActRate	The hourly network flow ratio of sWanIPAct to LanIP; the higher the value is, the higher the ratio is.
	ActWeight	Significant level of persistent communication for the connection from sWanIPAct to LanIP; the higher the value is, the more significant it is.
LanIP2sWanIPFail	FailHour	Time of communication for the connection from LanIP to sWanIPFail; the higher the value is, the longer the time is.
	FailRate	The hourly network flow ratio of sWanIPFail to LanIP; the higher the value is, the higher the ratio is.
	FailWeight	Significant level of persistent communication for the connection from LanIP to sWanIPFail; the higher the value is, the more significant it is.
	FailFlow	The number of failures of connection from LanIP to sWanIPFail; the higher the value is, the more connection failures there are.
	FailMatch	The number of connections between multisWanIPFail; the higher the value is, the more connections there are.
LanIP2IPPort	NoExist	The number of Lan IPs (LanIPs) of which the connection does not exist. Lan IP cannot route on Internet and it is used in an organization. One Lan IP corresponds to one client. Sometimes, LanIP sends messages to the unassigned Lan IP that does not exist in the organization. The larger the value is, the higher the number of unavailable connections is.
	DPortSum	The number of used destination Port of LanIP; the higher the value is, the higher the use frequency is.

Table XII. The Number of Network Flows of Each Hour (sWanIPAct to LanIP$_i$)

Source IP	Destination IP	Time Period (0–23)											fr_{ij}	$start_{ij}$	$null_{ij}$
		0	1	2	3	4	5	6	7	8	9	10-23			
LanIP$_i$	sWanIPFail$_1$		2	1	null	3	4	3	null	1	5		7	1	2
LanIP$_i$	sWanIPFail$_2$					5	3		null	1			3	4	1
LanIP$_i$	sWanIPFail$_3$					1		1	null	2	4		4	4	1
LanIP$_i$	∀sWanIPFail		2	1	null	9	7	4	null	4	8				

$\uparrow$

$end_i = 9$

- $\boldsymbol{null_{ji}}$ = the number of hours for no flow data $start_{ij}$ between the start and the end of the network activities, of which the communication is from all suspicious active Wan IPs (sWanIPAct) to the ith Lan IP (LanIP$_i$)

(3) **ActWeight = {ActWeight$_1$, ActWeight$_2$, . . . , ActWeight$_i$}**;
ActWeight$_i$ = max(aw_{1i}, aw_{2i}, . . . ,aw_{ji}), the i$_{\text{th}}$ feature value of LanIP;
$\boldsymbol{aw_{ji}}$ = $[log_{10}(ah_{ji})](ar_{ji})$, the weight of time for the connection from the jth suspicious active Wan IP (sWanIPAct$_j$) to the ith of Lan IP (LanIP$_i$)

(4) **FailHour = {FailHour$_1$, FailHour$_2$, FailHour$_i$}**
FailHour$_i$ = max(fh_{i1}, fh_{i2}, . . . , fh_{ij}), the ith feature value of LanIP;

Table XIII. ActRate Calculation

Source IP	Destination IP	fr_{ij}
LanIP$_i$	sWanIPFail$_1$	$\frac{7}{((9-1)+1)-2} = \frac{7}{7} = 1$
LanIP$_i$	sWanIPFail$_2$	$\frac{3}{((9-4)+1)-1} = \frac{3}{5} = 0.6$
LanIP$_i$	sWanIPFail$_3$	$\frac{4}{((9-4)+1)-1} = \frac{4}{5} = 0.8$

$\longrightarrow 1 = \max(1, 0.6, 0.8)$

$=\text{FailRate}_i$

***fh*$_{ij}$ =** the number of hours that the hourly network traffic flows connect from the ith Lan IP (LanIP$_i$) to the jth suspicious inactive Wan IP (sWanIPFail$_j$), which is greater than 0

(5) **FailRate = {FailRate$_1$, FailRate$_2$, ... , FailRate$_i$}**
FailRate$_i$ = max(fr_{i1}, fr_{i2}, ... , fr_{ij}), the ith feature value of LanIP;
$fr_{ij} = \frac{fh_{ij}}{(end_i - start_{ij} + 1) - null_i}$, the ratio of the number of hours of the jth suspicious inactive Wan IP (sWanIPFail$_j$) to that of the ith Lan IP (LanIP$_i$)

- ***end*$_i$** = the last hour of the network activity from the ith Lan IP (LanIP$_i$) to all suspicious inactive Wan IPs (sWanIPFail)
- ***start*$_{ij}$** = the first hour of the network activity from the ith Lan IP (LanIP$_i$) to the jth suspicious inactive Wan IP (sWanIPFail$_j$)
- ***null*$_{ij}$** = the number of hours for no flow data after $start_{ij}$ between the start and the end of the network activities, of which the communication is from the ith Lan IP (LanIP$_i$) to all suspicious inactive Wan IPs (sWanIPFail)

(6) **FailWeight = {FailWeight$_1$, FailWeight$_2$, ... , FailWeight$_i$}**
FailWeight$_i$ = max(fw_{i1}, fw_{i2}, ... , fw_{ij}), the ith feature value of LanIP;
$fw_{ij} = [log_{10}\,(fh_{ij})](fr_{ij})$, time weight of connection from the ith Lan IP (LanIP$_i$) to the jth suspicious inactive Wan IP (sWanIPFail$_j$))

(7) **FailFlow = {FailFlow$_1$, FailFlow$_2$, ... , FailFlow$_i$};**
FailFlow$_i$ = max(ff_{i1}, ff_{i2}, ... , ff_{ij}), the ith feature value of LanIP;
ff_{ij} = the total number of network flows from the ith Lan IP (LanIP$_i$) to the jth suspicious inactive Wan IP (sWanIPFail$_j$)

(8) **FailMatch = {FailMatch$_1$, FailMatch$_2$, ... , FailMatch$_i$}**
FailMatch$_i$ = the ith feature value of LanIP;
FailMatch$_i$ = the count of the ith LAN IP (LanIP$_i$) connecting to suspicious inactive Wan IP (sWanIPFail)

(9) **NoExist = {NoExist$_1$, NoExist$_2$, ... , NoExist$_i$}**
NoExist$_i$ = the ith feature value of LanIP;
NoExist$_i$ = the count of the ith Lan IP (LanIP$_i$) connecting to inexistent Lan IP; Section 2.2 mentioned that IP refers to all IP addresses. First, the Lan IP addresses are selected out from all IPs, and then the total number of inexistent Lan IPs is calculated.

(10) **DPortSum = {DPortSum$_1$, DPortSum$_2$, ... , DPortSum$_i$}**
DPortSum$_i$ = max(cp_{i1}, cp_{i2}, ... , cp_{ij}), the ith feature value of LanIP;
cp_{ij} = the count of the ith Lan IP (LanIPi) connecting to the used destination port of the jth IP

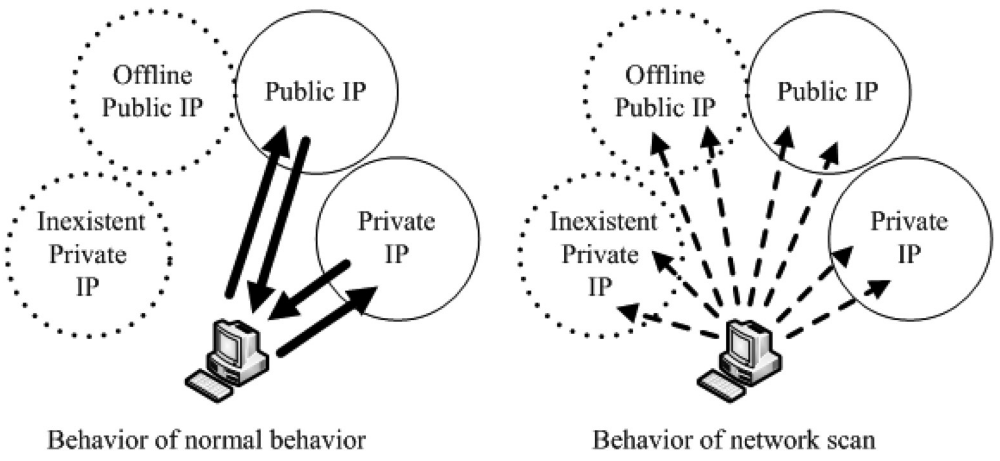

Fig. 5. Normal behaviors versus scanning behaviors.

3.4. Combine Same Behavioral Features to Form a Single Behavior: Behavior Layer

These 10 features generated in the LanIP Layer cannot represent all behaviors in Figure 2. In addition, the effect of clustering IPs by these features may not prove to be efficient. In the Behavior Layer, the same feature values are combined into one behavior. The behavior value is then used to measure the significant level of the behavior. The higher the value is, the more significant it is. The way to generate the behavior value is to normalize each feature value first, and then to accumulate the normalized value for each behavior. The details are elaborated as follows.
(1) Assign Relevant Features to a Behavior

- **ActBehavior**
  For the connection from the suspicious active Wan IP to Lan IP (LanIP), there are three time feature values, including ActHour, ActRate, and ActWeight. The ActBehavior of this proposed mechanism stands for the behavior of the long communication with all suspicious, outside network hosts. To this end, these three feature values belong to ActBehavior.
- **FailBehavior**
  FailHour, FailRate, and FailWeight are the time feature values that the Lan IP (LanIP) failed to connect to the suspicious inactive Wan IP (sWanIPFail). FailBehavior of this proposed mechanism is defined as the behavior of continuous connection failure with all suspicious outside network servers, and therefore, these three feature values belong to FailBehavior. FailFlow is another feature value of FailBehavior, which represents the number of connection failures. As for the successful connection, the count feature is ignored in this study, since it usually focuses on the time and not the number of connections.
- **ScanBehavior**
  DPortSum, FailMatch, and NoExist are the main feature values of ScanBehavior. In normal cases, client hosts only connect to the certain and often used destination port, except when conducting port scanning. Therefore, DPortSum can be used to predict the port-scanning behavior according to the total number of used ports. The left image of Figure 5 shows that the normal client can only access the existent IP. Once the unusual scanning behavior occurs, as the right image of Figure 5 shows, the client host will thus send all Wan IPs and Lan IPs, including the offline servers (Wan IPs) and the inexistent hosts (Lan IPs), a large number of messages by scanning the

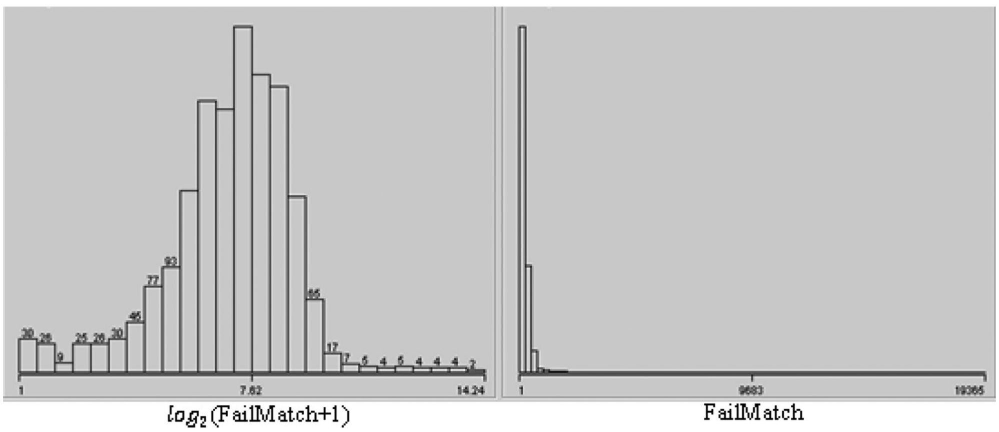

Fig. 6. Data distribution of FailMatch.

IP or Port, and also the inactive servers will respond with failure messages to the client host so that the abnormal scanning behavior will be discovered by observing the frequency of the failure information. Form the aforementioned discussion, FailMatch and NoExist are regarded as the feature values of ScanBehavior.

(2) Normalize the Feature Value

After analyzing and classifying the features of the behavior, the feature values of this proposed mechanism are required to be normalized using a min-max normalization procedure, as is shown in Equation (3.1).

$$\mathbf{v\_new} = \mathbf{v}/\mathbf{max\_a}, \tag{3.1}$$

where **v** is the value of each feature, **max_a** is the largest value of the feature, and **v_new** is the normalized value of the feature.

There exists a great data gap in the values of each feature, including that of FailFlow, FailMatch, NoExist, and DPortSum; Figure 6 mainly displays the data distribution of the feature FailFlow. The horizontal axis indicates the size of the data, and the vertical axis represents the number of the corresponding Lan IP (LanIP). The data shown in the right part of Figure 6 are the actual data. The variables with higher values take only a smaller part of the data as a whole; therefore, they broaden the distribution of the data, which is the major reason that there are difficulties in reading and analyzing the data. To improve this situation further, it is required to add 1 to the feature value and then to calculate its logarithm with a base 2, that is, ($log_2, i.e.,$ thefeaturevalue$+1$). The results are shown in the left part of Figure 6, which become the readable information of LanIP.

(3) Summarize the Normalized Feature Scores of the Same Behavior

N() represents the function of data normalization; the computations of ActBehavior, FailBehavior, and ScanBehavior are defined as follows:

ActBehavior = N(ActHour) + N(ActRate) + N(ActWeight)

FailBehavior = N(FailHour) + N(FailRate) + N(FailWeight) + N(log_2(FailFlow + 1))

ScanBehavior = N(log_2(FailMatch + 1)) + N(log_2(NoExist + 1)) + N(log_2(DPortSum + 1))

Table XIV shows the calculation results of these behaviors of LanIP.

Table XIV. Behaviors of IP from LanIP

LanIP	ActBehavior	FailBehavior	ScanBehavior
$LanIP_1$	2	1.5	0.7
$LanIP_2$	0.8	2.3	1.1
	~		
$LanIP_i$	0.8	1.8	0.9

3.5. Separate Bot Clients Out from Normal Clients: Cluster Layer

Higher values of ActBehavior, FailBehavior, and ScanBehavior of the Behavior Layer can be used to explain more about network behaviors, but the critical value of the network behavior of Botclient is usually hard to determine. Therefore, this study applies an approach of automatically merging the similar behaviors based on the principle of data clustering. If those values are low, this behavior, namely, a group of normal behaviors, does not meet the condition of being the three network behaviors of Botclient. In contrast, if the three values are low, this group of data meets the characteristics of these network behaviors and is detected and characterized as a group of abnormal behaviors.

The first step is to conduct the min-max normalization procedure [Visalakshi et al. 2009] to eliminate any possible data inconsistency. The normalized data should be in the range of 0 to 1 according to Equation (3.1). The second step is to cluster a group of data by using a hybrid PSO+K-means clustering algorithm [Merwe et al. 2003] that can generate more compact clustering results [Cui et al. 2005; Rana et al. 2010]. Finally, the Lan IP data are clustered into two groups, including a normal group with a small cluster centroid and an abnormal group with a large centroid. The Lan IP addresses in the abnormal group meet the characteristics of three network behaviors of Botclient, and therefore, those IP addresses are all Botclients. This process is shown in Figure 7.

The flow process of the Cluster Layer is described as follows:

Step 1. Normalize each behavior value/dimension of the Lan IP according to Equation (3.1).

Step 2. Decide the number of particles; randomly assign each particle a cluster centroid (particle position) and the movement of PSO.

Step 3. Each particle has to

(1) Calculate the fitness value according to Equation (3.1)

$$\text{Fitness} = \frac{\sum_{x=1}^{K}\left\{\frac{\sum_{\forall l_i \in A_x} d(c_x,l_i)}{s_x}\right\}}{CN} \tag{3.2}$$

Related variables:
l_i = the vector of the ith Lan IP, $l_i = \{l_1, l_2, l_i\}$
A_x = all vectors of Lan IP in cluster x
c_x = the vector of cluster center in cluster x
$d(c_x, l_i)$ = the distance between c_x and l_i
s_x = the amount of data in cluster x
K = the number of clusters

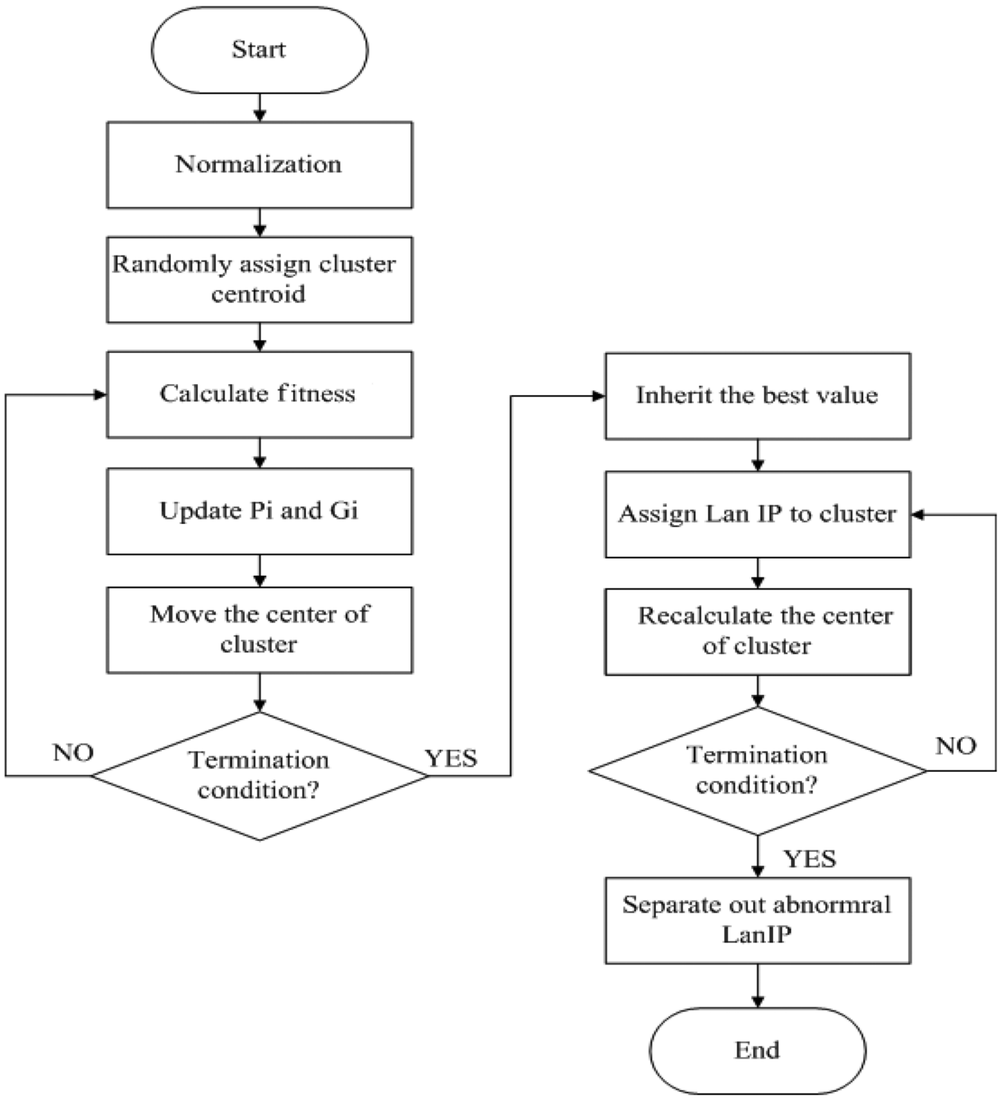

Fig. 7. The flow process of cluster layer.

(2) Update the best value of each particle.

Step 4. Assign the best value of each particle with the current fitness value.
Step 5. Move the location of the center of each cluster.
Step 6. Return to Step 3 and repeat until the end of all procedures.
Step 7. Inherit the center of the cluster that generated the best value.
Step 8. Assign each Lan IP the closest cluster center.
Step 9. Recalculate the center of each cluster.
Step 10. Return to Step 5 and continue the process until the current cluster center stops at the same position of the last center.

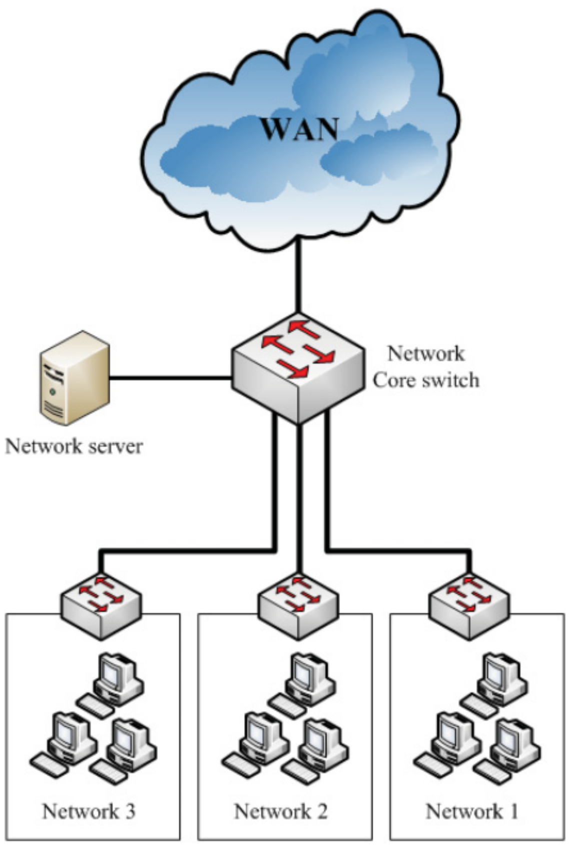

Fig. 8. Network architecture.

Step 11. Classify normal-data group and abnormal-data group according to the value of the cluster center; the abnormal one has smaller values; the Lan IP in the abnormal-data group is the Botclient that the proposed approach detected.

4. EXPERIMENTS AND EVALUATIONS

This study collects the real-world network traffic tracing from a campus network. The network architecture is shown in Figure 8. The flow records of the core switch are transferred into the server of network flows based on the Cisco NetFlow Protocol, and then the conversion programs are employed to import the flow records into the database. The specification of the server is shown in Table XV, and the network traffic records of 7 days are provided in Table XVI.

The analysis process adopts the network traffic records on March 21, 2012, as an example. This study keeps the best fitness value of each generation in the process of PSO (Figure 9). The horizontal axis represents a generation number, and the maximum reaches 100 generations; the vertical axis is the fitness value (the smaller the better). The optimization process starts to converge from the 30th generation.

4.1. Analysis Results I

This study analyzes the network flows from March 21 to March 27 and detects the daily Botclients during this period. Table XVII shows the daily number of Botclients

Table XV. Network Server Environment

CPU	Inter Core 2 Quad CPU Q9400 2.66GHz
Memory	8GB
Operation system	CentOS 5.6
Database	MySQL 5

Table XVI. Network Flow Records

Date	Data Count	Data Size(GB)
03/21/2012	15,028,425	2.3
03/22/2012	17,139,221	2.5
03/23/2012	11,112,617	1.7
03/24/2012	3,977,916	0.8
03/25/2012	6,414,370	1.8
03/26/2012	17,180,657	2.9
03/27/2012	16,524,514	2.5
Total	87,377,720	14.5

Table XVII. The Number of Botclients

Date	Total Number of Lan IP Addresses	Botclient (Based on This Mechanism)
03/21/2012	2,083	509
03/22/2012	2,106	575
03/23/2012	1,890	762
03/24/2012	568	199
03/25/2012	766	378
03/26/2012	2,094	601
03/27/2012	2,166	862

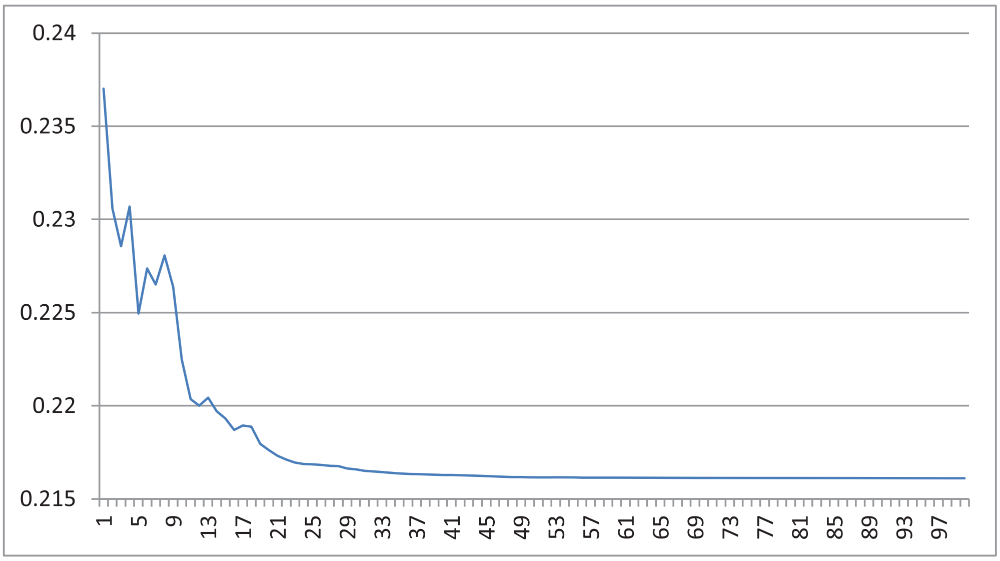

Fig. 9. The best fitness value of each generation.

detected by this proposed mechanism, and around 30% of Lan IP addresses are Botclients. The usage rate of the Internet on March 24 and March 25 was lower because of weekends, and for this reason, the number of Botclients was also fewer.

The Botclients uncovered by the proposed mechanism in this study include the abnormal events detected by the IDS and the Lan IP connecting to the Blacklist

Table XVIII. Statistics of the Number of Botclients

Date	This Mechanism	IDS & Blacklist	By IDS	Connecting to Blacklist
03/21/2012	509	369	173	337
03/22/2012	575	390	124	371
03/23/2012	762	460	80	440
03/24/2012	199	125	25	115
03/25/2012	378	255	78	247
03/26/2012	601	396	111	381
03/27/2012	862	578	141	557

Table XIX. Event Statistics of the Intrusion Detection System

Event Name	Count
ET MALWARE Suspicious User Agent (Autoupdate)	397
ET MALWARE Suspicious Mozilla User-Agent - Likely Fake (Mozilla/4.0)	137
ET MALWARE Mozilla User-Agent (Mozilla/5.0) Inbound Likely Fake	93
ET TROJAN Downadup/Conficker A or B Worm reporting	49
ET TROJAN IMDDOS Botnet User-Agent STORMDDOS	48
ET MALWARE User-Agent (User-Agent Mozilla/4.0 (compatible))	32
ET MALWARE AskSearch Toolbar Spyware User-Agent (AskTBar)	27
DNS:tacoda.at.atwola.com, Kaspersky:HEUR:Trojan.Win32.Generic	22
ET MALWARE Baidu.com Spyware Bar Pulling Data	16
ET TROJAN Suspicious User Agent Ryeol HTTP Client Class	14

(Table XVIII). For example, among the traffic flow records of 3/21, there are 509 Botclients being uncovered, of which 369 Botclients are successfully detected by the IDS and also have ever connected with the IP in the Blacklist. Specifically, out of 369 Botclients, there are 173 Botclients being detected by the IDS and 337 Botclients having ever connected to the Blacklist.

Table XIX summarizes the top 10 Botclient events occurring from March 21 to March 27 and lists separately the number of Botclients uncovered by the IDS. The largest share of all events is “ET MALWARE Suspicious User Agent (Autoupdate),” which is composed of 397 Botclients.

Figure 10 shows the top 10 countries that have had the connections from these Botclients, of which the mainland of China (cn) ranks first with 80.4%, and the United States (us) is in second place with 17.6%.

Figure 11 illustrates the data results of three behaviors obtained on March 21, 2012, using PSO and the K-means clustering algorithm. These results are plotted in a 3-dimensional space. The data marked with an “x” are attributed to an abnormal-data group, and each of them corresponds to a Lan IP address of Botclient. On the contrary, the data marked with an “o” are attributed to a normal-data group, consisting of Lan IP addresses of the general, client computers. By comparison of the abnormal-data group with the normal-data one, the former (i.e., Botclient) has more dimensions and its distribution of data is more dispersed.

4.2. Analysis Results II

If the detected Botclients based on this mechanism have never been uncovered by the IDS or have not connected onto the Blacklist on that day, they would be detected successively in the next few days and also build up the connections with the Blacklist. The results are further discussed next.

The Botclients of March 21 have not yet been located by the other systems, then on the next 6 days:

Country Code		Ratio
cn		80.4%
us		17.6%
Other	ru	0.45%
	tw	0.37%
	kr	0.24%
	my	0.2%
	vg	0.2%
	de	0.16%
	bs	0.12%
	ua	0.12%

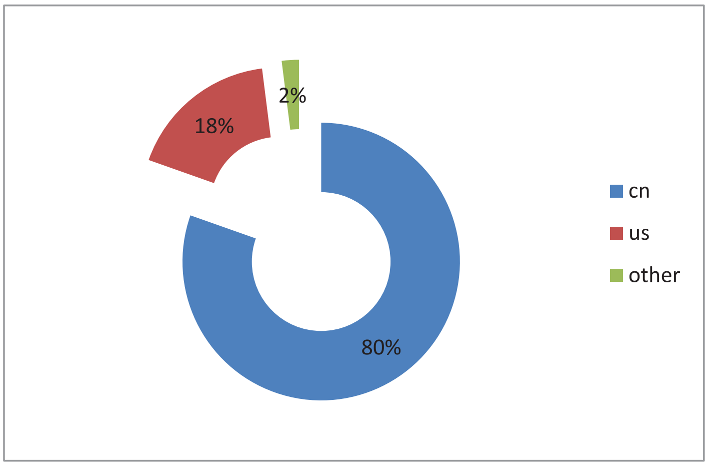

Fig. 10 Countries of the world in Blacklist.

On 3/22, 30 Botclients are detected by the IDS, and 59 Botclients connect to the Blacklist.
On 3/23, five Botclients are detected by the IDS, and 18 Botclients connect to the Blacklist.
On 3/24, one Botclient is detected by the IDS, and three Botclients connect to the Blacklist.
On 3/25, five Botclients are detected by the IDS, and 10 Botclients connect to the Blacklist.
On 3/26, 10 Botclients are detected by the IDS, and 15 Botclients connect to the Blacklist.
On 3/27, seven Botclients are detected by the IDS, and 21 Botclients connect to the Blacklist.

The Botclients of 3/22 have not yet been located by the other systems, then on the next 5 days:

On 3/23, eight Botclients are detected by the IDS, and 42 Botclients connect to the Blacklist.
On 3/24, four Botclients are detected by the IDS, and four Botclients connect to the Blacklist.
On 3/25, 13 Botclients are detected by the IDS, and 15 Botclients connect to the Blacklist.
On 3/26, 21 Botclients are detected by the IDS, and 34 Botclients connect to the Blacklist.
On 3/27, 13 Botclients are detected by the IDS, and 32 Botclients connect to the Blacklist.

The Botclients of 3/23 have not yet been located by the other systems, then on the next 4 days:

On 3/24, 11 Botclients are detected by the IDS, and 34 Botclients connect to the Blacklist.
On 3/25, 26 Botclients are detected by the IDS, and 60 Botclients connect to the Blacklist.
On 3/26, 26 Botclients are detected by the IDS, and 119 Botclients connect to the Blacklist.
On 3/27, 21 Botclients are detected by the IDS, and 94 Botclients connect to the Blacklist.

The Botclients of 3/24 have not yet been located by the other systems, then on the next 3 days:

On 3/25, 11 Botclients are detected by the IDS, and 10 Botclients connect to the Blacklist.
On 3/26, five Botclients are detected by the IDS, and 24 Botclients connect to the Blacklist.
On 3/27, three Botclients are detected by the IDS, and 12 Botclients connect to the Blacklist.

The Botclients of 3/25 have not yet been located by the other systems, then on the next 2 days:

On 3/26, 18 Botclients are detected by the IDS, and 56 Botclients connect to the Blacklist.
On 3/27, nine Botclients are detected by the IDS, and 24 Botclients connect to the Blacklist.

The Botclients of 3/26 have not yet been located by the other systems, then on the next day:

On 3/27, 33 Botclients are detected by the IDS, and 98 Botclients connect to the Blacklist.

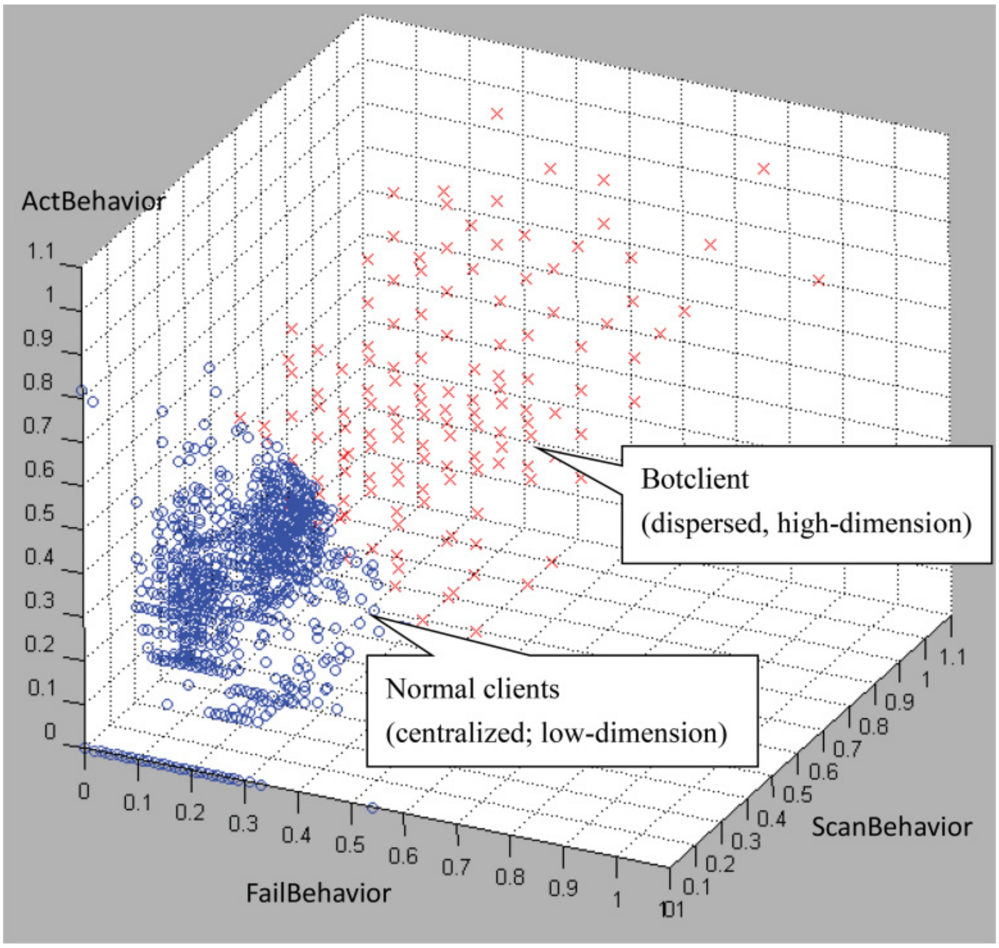

Fig. 11. Results of data clustering in three dimensions.

The numbers of Botclients of these aforementioned analyses are summarized and demonstrated in Figures 12 and 13. In summary, these experimental results show that the proposed mechanism does have an outstanding advantage to locate the Botclients earlier than the detection of/from other systems.

Figure 12. The number of Botclients (from March 21 to March 26) detected by the IDS.

Figure 13. The number of Botclients (from March 21 to March 26) detected by the BlackList.

5. CONCLUSIONS AND FUTURE WORKS

Most studies have used the packet contents or the network flow characteristics to conduct network analysis. Such an analysis approach has its weakness in that the contents and flow characteristics could be easily changed to avoid being spotted by the detection systems. In addition, it becomes rather difficult to analyze these packets, which are encrypted. Therefore, this study proposed a simple and feasible method that uses network behaviors, including ActBehavior, FailBehavior, and ScanBehavior, to detect the Botnet behaviors and further applies PSO+K-means clustering to locate

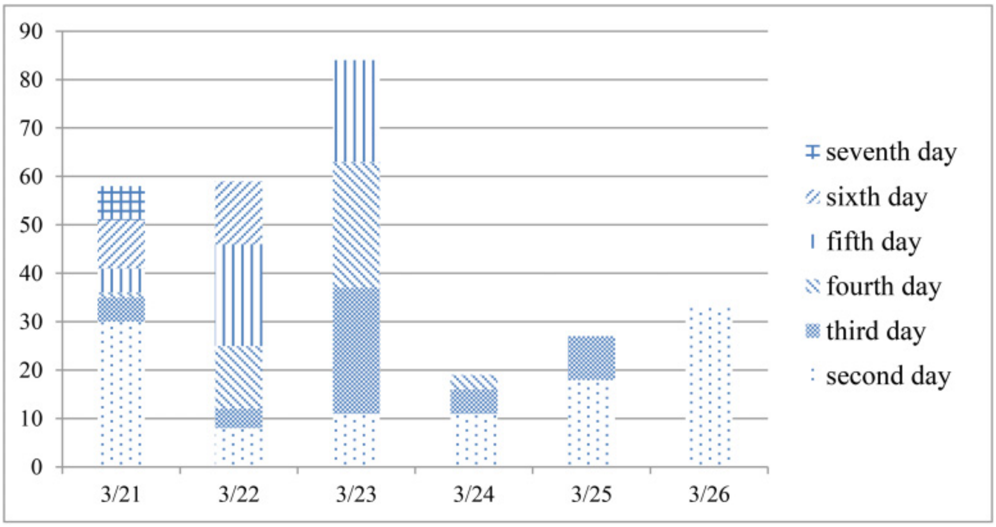

Fig. 12. The number of Botclients (from March 21 to March 26) detected by the IDS.

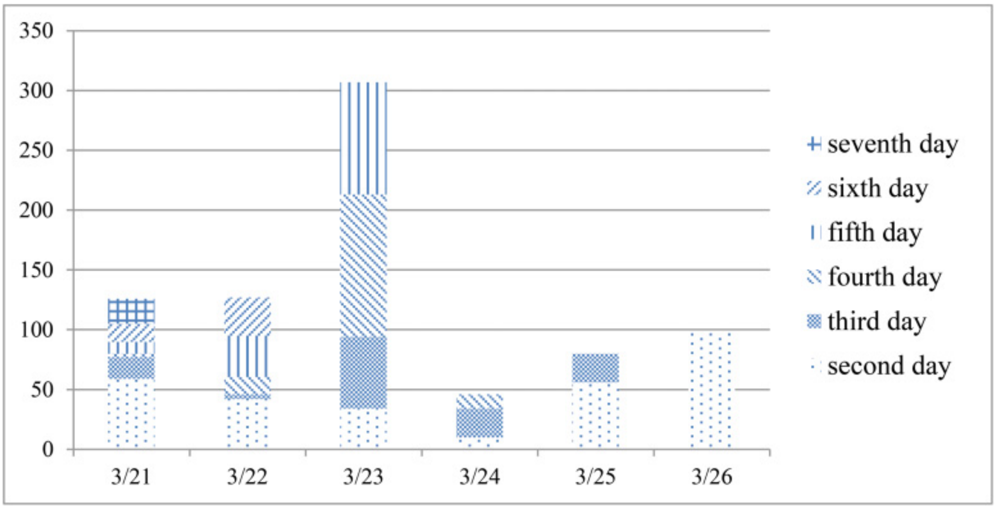

Fig. 13. The number of Botclients (from March 21 to March 26) detected by the BlackList.

the Botnet-infected client computers (Botclient). Because three behaviors always exist, this proposed detection approach proves to be more effective and efficient regardless of whether the packet contents or the network flow characteristics have been updated.

There exist three key issues remaining for the packet analysis. First, the packets' contents are related to the issue of data privacy and also increasing computing costs [Zeng et al. 2010]. Second, the encrypted contents are much more difficult to analyze. Third, packet characteristics can be easily changed by adding some extra characters so that it becomes difficult to locate the potential attacking host through the analysis of the packet contents. This study provides an effective solution to resolve these aforementioned problems. It uses indirectly the flow characteristics of the Network Layer and the Transport Layer to analyze the packets and to locate the Botclient. This proposed mechanism gets and defines the three main network behaviors occurring in the client hosts from the Network Layer first and then extracts the relevant patterns of each behavior from the Network Layer and the Transport Layer. Finally, the related patterns are joined into a network behavior.

This study accomplished the following purposes. First, the infected Botnet client hosts can be detected effectively. This study used the real network flow records of a university to analyze and locate the unusual Botclients based on this proposed mechanism. Those computers with such problems can also be well detected further by other detection systems and be linked to the Blacklist. Second, the proposed mechanism does not directly use the packet contents to perform the resulting analysis so that the encrypted packets do not affect the actual results of Botnet detection, and by operation, it is more effective than the previous approaches for analyzing the packet contents to locate Botclients. In fact, this study shows that the changes of flow characteristics have no effect on the three behaviors of Botclients based on this proposed mechanism. Third, and finally, this approach can recognize the Botclient that have not yet been detected by other detection systems and does not connect to the hosts in the Blacklist. Soon after, the other detection tools or systems can also uncover and catch this Botclient.

In summary, this mechanism provides a simple and straightforward method to locate the Botclient and does not require the installation of the various detection applications on the users' computers. Therefore, it is suitable for application to a dormitory network, a home network, and a mobile 3G network. In addition, the user's privacy is well protected, because the approach proposed in this study uses the traffic flows, rather than the decapsulated packet contents, to locate the suspicious Botclients.

This study uses the three elementary network behaviors of Botclient—ActBehavior, FailBehavior, and ScanBehavior—and applies the PSO+K-means clustering algorithm to predict the potential members of Botnet. The future works can be extended to explore the other behaviors of Botclient and further combine them with the other algorithms or use other approaches to improve the detection efficiency of this proposed mechanism.

ACKNOWLEDGMENTS

The authors also gratefully acknowledge the helpful comments and suggestions from the reviewers.

REFERENCES

B. Acohido and J. Swartz. 2009. Hacker attack takes down Twitter, Facebook, LiveJournal. Retrieved from http://www.usatoday.com.

A. Ahmadyfard and H. Modares. 2008. Combining PSO and k-means to enhance data clustering. In *Proceedings of the 2008 International Symposium on Telecommunications (IST'08)*. 688–691.

Y. Al-Hammadi, U. Aickelin, and J. Greensmith. 2008. DCA for bot detection. In *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI'08)*.

F. van den Bergh and A. P. Engelbrecht. 2006. A study of particle swarm optimization particle trajectories. *Information Sciences* 176, 8, 937–971.

S. Benson Edwin Raj and R. Shalini. 2012. A novel approach for the early detection and identification of botnets. *Advanced Materials Research* 403, 4469–4475.

N. Brownlee, C. Mills, and G. Ruth. 1999. Traffic flow measurement: Architecture. IETF. *RFC 2722.* 9–12.

J. Carr. 2009. *Inside Cyber Warfare.* O'Reilly Media.

X. Cui and T. E. Potok. 2005. Document clustering analysis based on hybrid PSO+K-means algorithm. *Journal of Computer Sciences* (Special Issue), 27–33.

Z. Du, Y. Wang, and Z. Ji. 2008. PK-means: A new algorithm for gene clustering. *Computational Biology and Chemistry* 32, 4, 243–247.

R. Eberhart and J. Kennedy. 1995. A new optimizer using particle swarm theory. In *Proceedings of the 6th International Symposium on Micromachine Human Sciences*, 39–43.

F-Secure Labs. 2014. Mobile Threat Report Q1 2014.

M. Feily, A. Shahrestani, and S. Ramadass. 2009. A survey of botnet and botnet detection. In *Proceedings of the 3rd International Conference on Emerging Security Information, Systems and Technologies*. 268–273.

G. Geng, G. Xu, M. Zhang, Y. Guo, G. Yang, and C. Wei. 2012. The design of SMS based heterogeneous mobile botnet. *Journal of Computers* 7, 1, 235–243.

German Honeynet Project. 2005. Tracking botnets. Retrieved from http://www.honeynet.org/papers/bots.

G. Gu, P. Porras, V. Yegneswaran, M. Frog, and W. Lee. 2007. BotHunter: Detecting malware infection through ids-driven dialog correlation. In *Proceedings of the 16th USENIX Security Symposium.* 167–182.

G. Gu, J. Zhang, and W. Lee. 2008. BotSniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network & Distributed System Security Symposium (NDSS'08).*

G. Gu, R. Perdisci, J. Zhang, and W. Lee. 2008. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th USENIX Security Symposium*. 139–154.

F. Hacquebord. 2011. Esthost Taken Down—Biggest Cybercriminal Takedown in History. Retrieved from http://www.trendmicro.com.

J. Han and M. Kamber. 2001. *Data Mining: Concepts and Techniques.* San Francisco: Morgan Kaufmann.

J. A. Hartigan, and M. A. Wong. 1979. A K-Means clustering algorithm. *Journal of the Royal Statistical Society Series C* 28, 1, 126–130.

C. T. Huang, K. J. Han, and J. Perretta. 2011. Automatic selection of routers for placing early filters of malicious traffic. In *Proceedings of the Global Telecommunications Conference.* 1–5.

A. K. Jain. 2010. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters* 31, 8, 651–666.

A. Karim, S. A. A. Shah, and R. Salleh. 2014. Mobile botnet attacks: A thematic taxonomy. In *New Perspectives in Information Systems and Technologies* 2, 153–164.

J. Kennedy, R. Eberhart, and Y. H. Shi. 2001. *Swarm Intelligence.* Morgan Kaufmann.

J. Kennedy and R. Eberhart. 1995. Particle swarm optimization. In *Proceedings of the IEEE International Joint Conference of Neural Networks,* 4. 1942–1948.

J. Kennedy. 1997. The particle swarm: Social adaptation of knowledge. In *Proceedings of the Conference on Evolutionary Computation.* 303–308.

B. C. Kim and Y. W. Park. 2012. Security versus convenience? An experimental study of user misperceptions of wireless internet service quality. *Decision Support Systems* 53, 1–11.

Y. K. Lam, P. W. M. Tsang, and C. S. Leung. 2012. PSO-based K-Means clustering with enhanced cluster matching for gene expression data. *Neural Computing & Applications* 1–7.

Z. Li, Y. Li, and L. Xu. 2011. Anomaly intrusion detection method based on k-means clustering algorithm with particle swarm optimization. In *Proceedings of the International Conference of Information Technology, Computer Engineering and Management Sciences* 2, 1, 157–161.

Y. Li. 2012. Theories in online information privacy research: A critical review and an integrated framework. *Decision Support Systems* 54, 1, 471–481.

Y. Lin, N. Tong, M. Shi, K. Fan, D. Yuan, L. Qu, and Q. Fu. 2012. K-means optimization clustering algorithm based on particle swarm optimization and multiclass merging. *Advances in CSIE* 1, 168, 569–578.

W. Lu, G. Rammidi, and A. Ali Ghorbani. 2011. Clustering botnet communication traffic based on n-gram feature selection. *Journal of Computer Communications* 34, 3, 502–514.

V. D. Merwe and A. P. Engelbrecht. 2003. Data clustering using particle swarm optimization. *Congress on Evolutionary Computation.* 215–220.

L. Messerschmidt and A. P. Engelbrecht. 2004. Learning to play games using a PSO-based competitive learning approach. *IEEE Transactions on Evolutionary Computation* 8, 3, 280–288.

McAfee. 2014. McAfee Labs 2014 Threats Predictions.

M. M. Masud, T. M. Al-Khateeb, K. W. Hamlen, J. Gao, L. Khan, J. Han, and B. Thuraisingham. 2011. Cloud-based malware detection for evolving data streams. *ACM Transactions on Management Information Systems* 2, 3, 16–26.

NetWitness. 2010. NetWitness Discovers Massive ZeuS Compromise. Retrieved from http://www.netwitness.com.

G. Ollmann. 2009. Botnet Communication Topologies. White Paper of Damballa.

N. S. Pai and S. P. Chang. 2012. Real-time face recognition system using an integrating PSO and K-means algorithm. *Advanced Science Letters* 9, 1, 179–184.

P. Piskac and J. Novotny. 2011. Network traffic classification based on time characteristics analysis. *Masaryk University Faculty of Informatics*. 4–14.

S. Rana, S. Jasola, and R. Kumar. 2010. A hybrid sequential approach for data clustering using K-Means and particle swarm optimization algorithm. *International Journal of Engineering, Science and Technology* 2, 6, 167–176.

R. A. Rodriguez-Gomez, G. Macia-Fernandez, and P. Garcia-Teodoro. 2011. Analysis of botnets though life-cycle. In *Proceedings of the International Conference on Security and Cryptography.* 257–262.

M. R. Rostami, B. Shanmugam, and N. B. Idris. 2011. Analysis and detection of P2P botnet connections based on node behaviour. In *Proceedings of the World Congress on Information and Communication Technologies.* 928–933.

W. T. Strayer, R. Walsh, C. Livadas, and D. Lapsley. 2006. Detecting botnets with tight command and control. In *Proceedings of the 31st IEEE Conference on Local Computer Networks.* 195–202.

J. Sun, W. Chen, W. Fang, X. Wun, and W. Xu. 2012. Gene expression data analysis with the clustering method based on an improved quantum-behaved particle swarm optimization. *Engineering Applications of Artificial Intelligence* 25, 2, 376–391.

Symantec. 2010. Trends for 2009. Symantec Global Internet Security Threat Report, Volume 15.

Symantec. 2011. Trends for 2010. Symantec Global Internet Security Threat Report, Volume 16.

Symantec. 2012. Trends for 2011. Symantec Global Internet Security Threat Report, Volume 17.

Symantec. 2014. Internet Security Threat Report 2014, Volume 19.

H. Tiirmaa-Klaar, J. Gassen, E. Gerhards-Padilla, and P. Martini. 2013. Botnets, cybercrime and national security. In *Botnets.* 1–40.

R. Turner. 2014. Tackling the DDoS Threat to Banking in 2014. White Paper of Alamai.

N. K. Visalakshi and K. Thangavel. 2009. Impact of normalization in distributed k-means clustering. *International Journal of Soft Computing* 4, 4, 168–172.

K. Wang, C. Y. Huang, S. J. Lin, and Y. D. Lin 2011. A fuzzy pattern-based filtering algorithm for botnet detection. *Journal of Computer Networks* 55, 15, 3275–3286.

L. Whitney. 2009. Amazon EC2 cloud service hit by botnet, outage. Retrieved from http://news.cnet.com

Z. H. Zhan, J. Zhang, Y. Li, and S. H. Chung 2009, Adaptive particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics* 39, 6, 1362–1381.

Y. Zeng, X. Hu, and K. G. Shin, 2010. Detection of botnets using combined host- and network-level information. In *Proceedings of the 2010 IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'10)*. 291–300.

Z. Zhu, G. Lu, and Y. Chen. 2008. Botnet research survey. In *Proceedings of the Annual IEEE International Computer Software and Applications Conference*. 967–972.

Y. Zhang, Y. Wang, and L. Qi. 2010. Exploring a swarm intelligence methodology to identify command and control flow. In *Proceedings of the IEEE 2010 3rd International Workshop on Advanced Computational Intelligence (IWACI'10)*. 318–322.

D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, and D. Garant. 2013. Botnet detection based on traffic behavior analysis and flow intervals. *Computers & Security* 39, 2–16.

Received May 2013; revised August 2014, September 2014; accepted October 2014