2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing

BotCatch: A Behavior and Signature Correlated Bot Detection Approach

Yuede Ji and Yukun He
College of Computer Science and Technology,
Jilin University,
Changchun Jilin, China,
Email: {jiyd12, heyk12}@mails.jlu.edu.cn

Qiang Li and Dong Guo
College of Computer Science and Technology,
Jilin University,
Changchun Jilin, China,
Email: {li_qiang, guodong}@jlu.edu.cn

***Abstract*—Botnet has become one of the most serious threats to Internet security. Compared with network-based bot detection approaches, host-based approaches can discover more insights of unknown bots, and we may completely eliminate bots if we can successfully detect them on end-host. Host-based approaches mainly include signature and behavior-based detection approaches. In this paper we propose a behavior and signature correlated bot detection approach, BotCatch. Firstly, we present the design of BotCatch. There are four components in BotCatch: analysis engine, signature analysis engine, behavior analysis engine, and correlation engine. The analysis engine assigns the suspicious sample to signature analysis engine and behavior analysis engine. These two engines analyze the sample and generate signature and behavior analysis result. Then correlation engine correlates these two analysis results to generate the final detection result. There is also a feedback mechanism which presents the correlation result to behavior analysis engine to guide its learning procedure. Secondly, we analyze the effectiveness of our correlation approach compared with signature-based and behavior-based bot detection approach. The analysis indicate that our correlation approach can effectively improve the detection accuracy. Thirdly, we evaluate our approach through experiments and the result indicate that our approach can detect bots effectively.**

***Keywords*—bot detection; host based; behavior and signature; correlation; feedback;**

I. Introduction

Botnet has become one of the most serious threats to Internet security [1]. A bot is a host compromised by malwares under the control of the botmaster through Command and Control (C&C) channel (i.e., IRC, HTTP, P2P, etc.). A large scale of bots form a botnet. The botmaster can utilize botnet to conduct various cyber crimes such as spreading malwares, DDoS attacks, spamming, phishing, and other cyber crimes. Botnet has become the major platform for most online criminal activities.

In order to defend bots, a large number of researches have been carried on regarding the detection of bot and botnet. According to the execution location, existing approaches can be divided into three categories: host-based approaches, network-based approaches, and host and network correlation approaches. (1) Host-based approaches mainly include signature- and behavior-based detection approaches [2]. Signature-based detection approaches mainly extract the feature information of the suspicious program to match with a knowledge database of existing bot, such as Rishi [3]. Behavior-based detection approaches monitor the abnormal behaviors on hosts to determine whether the host is infected, such as the status of operating system, the running status of suspicious programs, access to suspicious Registries, files, system call sequences, etc. [4], [5], [6]. (2) Network-based approaches mainly analyze the network traffic to filter out the host which has abnormal traffics [7]. Network-based approaches will be ineffective with traffic encrypted, C&C protocols changed, etc. These approaches are not able to identify the malicious process. (3) Host and network correlation approaches correlate host information with network traffic to detect bots [8], [9]. This is a new detection mechanism at the exploration stage because existing approaches still have the problems of both host- and network- based approaches.

Compared with network-based approaches, host-based approaches can discover more insights of unknown bots, and more importantly, we may completely eliminate the bot if we can successfully detect it on end-host. In signature-based bot detection approaches, the known bot programs and suspicous programs do not need to run. Thus signature-based approaches have low risks. While behavior-based approaches need to run all of them, thus they have high risks. Signature-based approaches use the match method to generate the detection result, if the match result reaches a level the program is regarded as bot. Thus it can only detect known bots, to be more specific, it can only detect the bots in the knowledge database. While behavior-based approaches monitors the real execution behaviors to determine whether the programs are malicious. Thus they can detect unkown bots. Most signature-based approaches have low false positives because they are based on the strict match result. They have different false negatives according to the difference between experiment data and knowledge database. While in behavior detection approaches, up-to-date bot programs use many hidden mechanisms and act like benign programs. Thus behavior-based approaches have low detection accuracy. From the perspective of overhead, signature-based approaches only analyze the suspicious binary to extract its signature, while behavior-based approaches need to monitor all the system calls. Thus signature-based approaches have low overhead and behavior-based approaches have high overhead. Up-to-date bots use many obfuscation techniques to evade the detection, and these techniques can evade signature-based approaches while well-designed behavior-based approaches are still able to detect them. The comparison of these two approaches are summarized in Table I.

978-0-7695-5088-6/13 $31.00 © 2013 IEEE
DOI 10.1109/HPCC.and.EUC.2013.230

	Advantage	Disadvantage
Signature-based	Low risk Low false positives Low overhead	Unable to detect unknown bots Unable to deal with obfuscations Require a lot of prior knowledge
Behavior-based	Can detect unknown bots Can deal with obfuscations Real-time detection	High risk Low detection accuracy High overhead

TABLE I. COMPARISON OF SIGNATURE-BASED AND BEHAVIOR-BASED APPROACHES

Behavior-based and signature-based approaches have many opposite features as shown in Table I. Thus if we can combine these two approaches, we may overcome some critical limitations of each approach and generate a better bot detection solution. We design BotCatch, a behavior and signature correlated bot detection approach. There are four components in BotCatch: analysis engine, signature analysis engine, behavior analysis engine, and correlation engine. The analysis engine assign the suspicious sample to signature analysis engine and behavior analysis engine. These two engines analyze the sample and generate signature analysis result and behavior analysis result. Then correlation engine correlates these two analysis results to generate the final detection result. Our experiment result indicate that BotCatch can detect bot programs effectively.

Our work makes the following contributions:

(1) We design a bot detection approach called BotCatch. There are 4 components: analysis engine, signature analysis engine, behavior analysis engine, and correlation engine. BotCatch generates the final detection result through correlating signature analysis result with behavior analysis engine.

(2) We propose a feedback mechanism between behavior analysis engine and correlation engine. This feedback mechanism presents the correlation result and signature detection result to behavior analysis engine to guide its learning procedure.

(3) We evaluate our approach through three experiments: signature-based bot detection, behavior-based bot detection, and signature and behavior correlated bot detection approach. The results indicate that correlated bot detection approach can improve the detection accuracy.

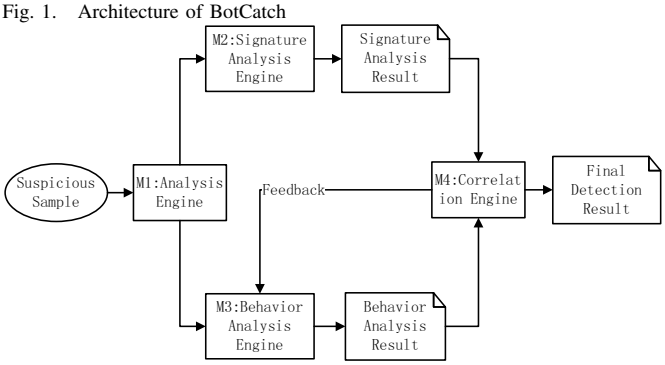
The remainder of this paper is outlined as follows: Section II presents the design of BotCatch in detail. Section IV presents the implementation and evaluation of BotCatch. Section III analyzes the effectiveness of BotCatch. Section V presents the limitation and future work. Section VI is related work. Section VII concludes this paper.

II. BOTCATCH DESIGN

A. System Architecture

As shown in Figure 1, BotCatch is composed of 4 components: analysis engine, signature analysis engine, behavior analysis engine, and correlation engine. The analysis engine assign the suspicious sample to signature analysis engine and behavior analysis engine. The behavior analysis engine analyzes the suspicious sample in a virtual environment and generates the behavior analysis result. The signature analysis engine analyzes the sample using online analysis tool virustotal [10] to generate the signature result. Then the correlation engine correlates these two analysis results to generate the final detection result. There is also a feedback mechanism between behavior analysis engine and correlation engine. This feedback mechanism provides the correlation result and signature detection result to behavior analysis engine to guide the learning procedure of behavior analysis engine.

Fig. 1. Architecture of BotCatch



B. Analysis Engine

Our analysis engine receives the suspicious samples submitted to BotCatch and then assign them to signature and behavior analysis engine. There are three submission ways, using the web utility, using the submission API of Python, and using the interface of smaple database. The easiest way is web while it is not easy to accomplish the automated analysis. The Python API also uses the interface of database, thus we directly use the interface of database to accomplish the automated analysis. The BotCatch also manages the assignment. If the signature and behavior analysis engine are in waiting state, the sample will be directly assigned to them. If they are analysing other sample, the new samples will be stored in our database. When they finish the analysis, the analysis engine will send the new sample to them according to the sequence of submission time.

C. Behavior Analysis Engine

Behavior analysis engine executes a suspicious sample in an isolated environment. Thus we can monitor the sample behaviors without modifying the guest OS or installing additional softwares, and reduce the risk of being detected by bot programs. There are five phases in botnet life cycle: initial injection, secondary injection, connection or rally, malicious activities, and maintenance and upgrading [2]. According to the behaviors of these 5 phases, we monitor the Registry behaviors, file behaviors, and network behaviors of bot programs to extract its behaviors. We monitor its Registry behaviors and file behaviors in the isolated environment, its network behaviors through monitoring the packets of virtual ethernet adapter in host. We extract 11 behavior features to generate the behavior feature vector as shown in Table II, including 4 Registry features, 3 file features, and 4 network features.

According to the feature vector, we use Support Vector Machine (SVM) to generate the behavior suspicious value b. SVM learns from benign and bot feature vectors to predict unlabeled behavior vectors. Many previous researches focused on the

TABLE II. BEHAVIOR FEATURE VECTOR

Index	Feature Description
1	Creation or Modification of AutoRun Key in Registry
2	Creation or Modification of Process Injection Key in Registry
3	Creation or Modification of Other Critical Registry Key
4	DLL Creation into System Directory
5	EXE Creation into System Directory
6	Modification of Files in System Directory
7	Creation of Other Files in System Directory
8	Number of Ports Opened
9	Number of Suspicious Ports
10	Number of Unique IPs Contacted
11	Number of Suspicious IP

binary (malicious or benign) classification and the results are likely to be inaccurate because of the learning procedure [8]. In order to improve the learning model, we calibrate the distance score to a posterior classification probability indicating how likely a test feature vector belongs to a particular class [11].

$$Pr(y=1|x) \approx P_{A,B}(f) \equiv \frac{1}{1+exp(Af+B)}, where f = f(x). \tag{1}$$

In order to calculate the posterior classification probability of new sample, we need to calculate A, B and f. We can get A, B through the training dataset. While $f(x)$ is the decision value of the sample vector, we can get it from the prediction procedure. From Equation 1, we can conclude that posterior classification probability is a value between 0 and 1. We use b to represent the behavior suspicion value.

D. Signature Analysis Engine

Signature-based bot detection approaches should build a large knowledge database, while we are lack of large scale bot samples. Antivirus scanner is a kind of malware detectors, and malware detector is a system that attempts to identify malware using signatures and other heuristics techniques [12]. Thus we can use the knowledge databases of antivirus companies to overcome this weakness. We use the interface of virustotal to get the signature analysis result. Virustotal is a free online service that analyzes files and URLs enabling the identification of viruses, worms, trojans and other kinds of malicious content detected by antivirus engines and website scanners [10]. Virustotal can return the detection results of about 47 different antivirus engines with the most updated signature database. Our signature analysis engine submits the suspicious sample to virustotal and uses the returned result for further analysis. Virustotal has 47 antivirus engines at present, and every engine returns the result of 0 or 1. 0 denotes it is benign and 1 denotes malicious. The antivirus companies are in different regions, and some bot samples may only be discovered in several regions. Thus we use the ratio of positives with the total number to denote the signature suspicious value. Suppose there are d antivirus engines detect it as malicious, thus the signature analysis result will be $d/47$. We use s to denote the signature suspicious value.

E. Correlation Engine

The input to our correlation engine is behavior analysis result and signature analysis result of a suspicious program. The behavior analysis result is the behavior suspicious value b and the signature suspicious value s. We use w to denote the final suspicious value as shown in Equation 2.

$$w = \begin{cases} s & , s \geq \beta \\ \alpha s + (1-\alpha)b & , s < \beta \end{cases} \tag{2}$$

Since $s = d/47$, $b \in [0,1]$, and $\alpha \in [0,1]$, w is also a value between 0 and 1. Suppose γ is the threshold, and if $w < \gamma$, the suspicious program is benign, malicious on the contrary. In this equation, firstly, we use signature suspicious value s to compare with β. If $s \geq \beta$ we will not consider the behavior suspicious value b. Because s is the signature suspicious value, if many antivirus engines detect it as malicious. That means this program is a known malicious program and we can detect it as malicious without behavior detection. We set β greater than γ and β is a value greater than 0.5. Suppose $\beta = 0.6$, if $s \geq \beta$, that means there are more than 28 antivirus engines detect this program as malicious. In this case we can detect it as malicious because β is greater than γ. If $s < \beta$, we will use the second formula to calculate w. In this formula, we set α as a value smaller than 0.5. Because in this case, behavior detection result plays a more important role and should has a higher weight. We can get the final suspicious value w after the correlation procedure, thus we can get the final detection result through comparing with γ.

Besides generating the final detection result, correlation engine has another effect, dynamic guiding the machine learning procedure of behavior analysis engine. If s is greater than β, and if behavior suspicious value is not well, suppose b is smaller than δ. Then we can add this feature vector to our training set. We will not retrain our model immediately because frequent training will cause heavy overheads and a small number of new samples makes no significance. Thus when the number of new samples reaches a certain threshold, such as a half of the trained samples, we can retrain our models. In this way, we can gradually make our behavior analysis more accurate without causing too much overheads. This dynamic learning procedure can make our behavior analysis adapt the constantly changes of bots.

III. EFFECTIVENESS ANALYSIS

Signature-based bot detection approach can detect known bots with a high detection accuracy, while it will be ineffective with variants or new bots. Behavior-based bot detection approach can detect both known and unknown bots with almost equal detection accuracy while not high enough. BotCatch can detect known bots with a high detection accuracy like signature-based bot detection approach, and unknown bots with a higher detection accuracy than behavior-based approach. We will compare these three approaches using assumption analysis.

Suppose the bot samples used for detection are composed of 50% well-known bots, 30% rare bots, 20% new bots. Well-known bots represent the bots discovered in many regions and many times. The rare bots represent the bots only discovered in few regions and few times. The new bots represent the unknown bots. In signature-based bot detection approach, suppose the detection accuracy of known bots is a value between $[s_1, s_2]$, the suspicion value of rare bots is a value between $[s_3, s_4]$, and the suspicion value of new bots is a value between $[s_5, s_6]$. In behavior-based bot detection approach, different bot samples have almost the same detection result.

Suppose the suspicion value is a value between $[b_1, b_2]$. In BotCatch, the final detection result is the equation in Equation 1.

TABLE III. DETECTION VALUES

	Known Bots(50%)	Rare Bots(30%)	New Bots(20%)
Signature Detection	$[s_1, s_2]$	$[s_3, s_4]$	$[s_5, s_6]$
Behavior Detection	$[b_1, b_2]$		
BotCatch	s or $\alpha s + (1-\alpha)b$		

We have defined γ as the final detection threshold, thus if the detection value is greater than γ it will be a bot sample, benign on the contrary. Signature-based detection approach can detect almost all the known bots, thus $\gamma < s_1 < s_2$. It can detect a part of the rare bots, thus $s_3 < \gamma < s_4$. It can barely detect the new bots, thus $s_5 < s_6 < \gamma$. Behavior-based detection approach can detect a part of bot samples, thus $b_1 < \gamma < b_2$. We suppose all the detection values are symmetrical distribution. Thus the detection accuracy of signature-based detection approach is $\frac{s_4-\gamma}{s_4-s_3} * 30\% + 50\%$. The detection accuracy of behavior-based detection approach is $\frac{b_2-\gamma}{b_2-b_1} * 100\%$.

In BotCatch, we defined another threshold β, if $s \geq \beta$, $w = s$, otherwise $w = \alpha s + (1-\alpha)b$. For the known bots, $\gamma < \beta < s_1 < s_2$. For other two bot kinds, suppose their detection value are below β. Thus the detection value is between $[\alpha s + (1-\alpha)b_1, \alpha s + (1-\alpha)b_2]$. For the rare bots, the detection value is between $[\alpha s_3 + (1-\alpha)b_1, \alpha s_4 + (1-\alpha)b_2]$. For the new bots, the detection value is between $[\alpha s_5 + (1-\alpha)b_1, \alpha s_6 + (1-\alpha)b_2]$. Thus the detection accuracy of BotCatch is $50\% + \frac{\alpha s_4 + (1-\alpha)b_2 - \gamma}{(\alpha s_4 + (1-\alpha)b_2) - (\alpha s_3 + (1-\alpha)b_1)} * 30\% + \frac{\alpha s_6 + (1-\alpha)b_2 - \gamma}{(\alpha s_6 + (1-\alpha)b_2) - (\alpha s_5 + (1-\alpha)b_1)} * 20\%$.

TABLE IV. DETECTION ACCURACY

	Detection Accuracy
Signature Detection	$50\% + \frac{s_4-\gamma}{s_4-s_3} * 30\%$
Behavior Detection	$\frac{b_2-\gamma}{b_2-b_1} * 100\%$
BotCatch	$50\% + \frac{\alpha s_4 + (1-\alpha)b_2 - \gamma}{(\alpha s_4 + (1-\alpha)b_2) - (\alpha s_3 + (1-\alpha)b_1)} * 30\%$ $+ \frac{\alpha s_6 + (1-\alpha)b_2 - \gamma}{(\alpha s_6 + (1-\alpha)b_2) - (\alpha s_5 + (1-\alpha)b_1)} * 20\%$

We assign some specific values to the variables to present a easy understanding. According to the values in Table V, we calculate the detection accuracy as shown in Table VI. In behavior detection, the detection value relies on the behaviors of different samples. The known bots may not perform suspicious behaviors, while new bots may perform. Thus the difference between different samples are not clear, and we only use the total detection accuracy to present. From the detection accuracy in Table VI, we can see that Botcatch performs as well as signature-based approach in detecting known bots. It performs better than signature-bsed approach in detecting rare bots and it can detect new bots in an acceptable value. The total detection accuracy presents that BotCatch can achieve a well detection accuracy in detecting bots.

TABLE V. ASSUMPTION VALUES

s_1	s_2	s_3	s_4	s_5	s_6	b_1	b_2	α	β	γ
0.7	0.9	0.4	0.6	0.1	0.2	0.3	0.8	0.2	0.6	0.5

TABLE VI. DETECTION ACCURACY USING SPECIFIC VALUES

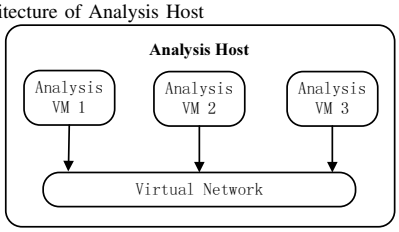
	Known Bots(50%)	Rare Bots(30%)	New Bots(20%)	Total
Signature	100%	50%	0%	65%
Behavior	-	-	-	60%
BotCatch	100%	59.1%	42.9%	76.3%

IV. EXPERIMENT

A. Implementation

The architecture of our analysis host is shown in Figure 2. The analysis host has three analysis virtual machines. They form an isolated local network. In order to capture the accurate network information, we configured the analysis machines to connect with the Internet. The analysis host has the following configurations: Intel Q6600 quad-core processor, 2.40GHz, 2GB RAM, and Ubuntu 12.04 operating system. We use VirtualBox 4.2.6 as our virtual machine with Windows XP SP3. We use cuckoo [13] as our analysis system. Cuckoo is the leading open source automated malware analysis system. We installed it on our analysis host. Our bot binaries are collected from [14], and there are more than 5 million malwares in their site. In behavior analysis, we use a library of SVM, libsvm [15].

Fig. 2. Architecture of Analysis Host



B. Experiment Results

We collect 625 binaries from [14], while about 237 samples are ineffective. Thus we analyze 388 samples with 338 bot samples and 50 benign samples. In order to evaluate the difference between correlated approach with signature-based and behavior-based approch, we make three experiments: signature-based bot detection, behavior-based bot detection, and behavior and signature correlated bot detection. We use false positive (FP) and false negative (FN) rates to present the result of our experiment. A false positive is defined as a benign program mistakenly detected as a bot program, and a false negative represents that a bot program is mistakenly detected as a benign program.

TABLE VII. EXPERIMENT RESULTS

	False Positive Rate	False Negative Rate
Signature Detection	0%	1.78%
Behavior Detection	16.7%	0%
Correlated Detection	5.56%	0%

In these experiments, we set $\alpha = 0.2$, $\beta = 0.6$, and $\gamma = 0.4$. In signature-based experiment, if the detection rate of the sample is greater than γ, it will be regarded as bot sample. That means, if a program is detected as malicious by more than 19 antivirus engines out of 47, it will be malicious. In this experiment, 332 bot samples are detected as malicious, 6 bot samples are detected as benign, and all benign samples are

detected as benign. The 0% false positive rate lies in our bot samples. Because these samples are known bot programs and their signatures are in the database of most antivirus engines. The reason of 6 undetected bot samples may be that these antivirus companies are in different regions and these bot samples may only be discovered in few regions.

TABLE VIII. BEHAVIOR-BASED EXPERIMENT SAMPLES

	Sample Numbers	Bot Samples	Benign Samples
Train Samples	345	302	43
Evaluation Samples	43	36	7
Total	388	338	50

In behavior-based bot detection experiment, we divide the samples into 10 groups, 9 of them are used for training, and the rest 1 group is for evaluation. That means 345 samples are used for training, including 302 bot samples and 43 benign samples. Other 43 samples are used for evaluating our approach, including 36 bot samples and 7 benign samples. The summaries are shown in Table VIII. The detection results are shown in Table VII, false positive rate is 16.7% and false negative rate is 0%. That means, 6 benign samples are detected as bot, and all bot samples are detected as bot. The reason of high false positive rate may be that the behaviors of benign programs are similar with bot programs, and the distribution of train samples are not well enough.

In behavior and signature correlated bot detection experiment, we also divide the samples like behavior-based bot detection as shown in Table VIII. The false positive rate is 5.56% and false negative rate is 0%. The correlation algorithm is shown in Equation 2. The false positive rate is higher than signature-based bot detection approach and lower than behavior-based. The false negative rate is 0%. The experiments indicate that correlated bot detection approaches can get a better detection result than signature-based detection approach and behavior-based detection approach.

V. LIMITATION AND FUTURE WORK

A. Limitation

This work presents a preliminary analysis of our research and several limitations exist in it: (1) We use virtual machine to run the suspicious program. Some advanced bot programs can detect whether they are running in a virtual or emulated environment. Also, there is an agent running in the virtual host. Some bot programs can detect these programs. This is a common problem of detection approaches using visualized or emulated techniques. While we can decrease this impact using a large scale of malicious samples. (2) The bot sample runs in the virtual machine for only several minutes. Thus many bot programs are terminated before they finish the life cycle of botnet, and they perform few specific bot behaviors. While this problem has no effect on signature analysis, we can decrease its impact through well-designed correlation engine. (3) Some special designed bots can use obfuscation techniques, multiprocess mechanism, response delay mechanism to hide the detection of our approach. (4) The distribution of bot samples in our experiment are not well enough, such as lacking of new generation of bots. We will try to add new bots in our future works. (5) The correlation engine in our approach is a primary component, the design of it determines the detection result. Our correlation algorithm is a little simple and not well enough to balance the weighs of signature and behavior in different situations. While the analysis and experiment can still prove the effectiveness of our correlation engine.

B. Future Work

According to the limitations and our research progress, our future work will focus on the following aspects: (1) The cuckoo analysis system still has many features that we have not completely understood yet, and we will continue study this analysis system. We know cuckoo has several limitations in analyzing bot programs, such as in a virtual machine, running only several minutes. We will try to fix these limitations in our future work. (2) The samples in our experiment have several limitations, the number of bot samples are not large enough, the distribution of them are not well. We will continue capturing bot samples, especially new generation of bots, in our future works. (3) The correlation algorithm in our correlation engine needs to be further optimized, the values of the coefficients and the threshold values should be careful selected. We will try to optimize these values through machine learning algorithms. (4) The feedback mechanism between behavior analysis engine and correlation engine will be further studied. This feedback mechanism makes the learning algorithm some intelligent.

VI. RELATED WORK

Ammar *et al.* propose a framework combining signature-based with behavior-based techniques using API graph system [12]. There are three procedures in their framework, preprocessing, graph construction, and graph matching. The preprocessing procedure execute the PE file and collect API call after unpacking complete. The graph construction procedure constructs the call graph based on API call and OS resource, and then decrease the constructed API graph. The graph matching procedure matches the graph with API call graph DB. This framework combines behavior and signature information while it is just a ideal framework without further analysis or experiments. Guo *et al.* propose a novel malware detection framework based on binary translation named HERO [16]. It is a novel framework that exploits static and dynamic binary translation features to detect broad spectrum malware and prevent its execution. They first use static binary translator based analyzer to analyze the binary file, if it can not complete the anlysis, they will use dynamic binary translator based analyzer to analyze the binary file. Their work still face the problem of low detection accuracy, low false alarm rate. Seungwon *et al.* propose the EFFORT [6], and there are 5 modules in their approach, human-process-network correlation analysis module, process reputation analysis module, system resource exposure analysis module, network information trading analysis module, and correlation engine. The correlation engine correlates the other modules detection result to generate the final detection result. Zeng *et al.* propose a host and network correlation approach to detect bot [8]. They use Process Monitor to monitor the information on host, including Registry, File System, and Network Stack. The suspicion level generator uses the extracted feature vector to generate the suspicion level. The router passes the collected Netflow of each time window to Flow Analyzer. Then it will analyze the Netflow, extract the feature vector, and pass it to the

cluster analyzer. It will cluster the hosts in LAN based on the network feature vector of each time window and the preprocessed information of host distance, and then pass the results to correlation engine. Through sending requests to all hosts in each cluster, the correlation engine will combine host information with network information to calculate the final detection result to determine whether the host has been infected. Hsiao *et al.* propose an approach combining dynamic passive analysis and active fingerprinting for bot detection in virtualized environments [19]. The passive detection agent lies in the virtual machine monitor to profile the bot behavior and check against it with other hosts. The active detection agent sends specific stimulus to a host and examine if there exists expected triggered behavior. While the active agent needs to know a lot of specific bot commands. Some online malware analysis system like Anubis [17], CWSandbox [18] use emulated environment to analyze submitted suspicious samples, after analysis, they will return a detailed analysis report to users. While our works are different from them, we focus on the detection of bot, and we correlate behavior analysis with signature analysis to generate a more accurate detection result, we also use correlation analysis result to guide the learning of behavior analysis engine.

VII. Conclusion

We propose a behavior and signature correlated bot detection approach, BotCatch. Firstly, we present the design of BotCatch. There are four components in BotCatch: analysis engine, signature analysis engine, behavior analysis engine, and correlation engine. The analysis engine assigns the suspicious sample to signature analysis engine and behavior analysis engine. These two engines analyze the sample and generate signature analysis result and behavior analysis result. Then correlation engine correlates these two analysis results to generate the final detection result. There is also a feedback mechanism which presents the correlation result and signature detection result to behavior analysis engine to guide the learning procedure of behavior analysis engine. Secondly, we analyze the effectiveness of our correlation approach compared with signature-based bot detection and behavior-based bot detection approach. The analysis indicate that our correlation approach can effectively improve the detection accuracy. Thirdly, we evaluate our approach through experiments and the result indicate that our approach can detect bots effectively.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant No.61170265, Fundamental Research Fund of Jilin University under Grant No. 201003035, 201103253. The authors would like to thank the team of cuckoo for their kind and generous to make their system open.

Corresponding author: Qiang Li, Email: li_qiang@jlu.edu.cn

References

[1] Abdullah, Raihana Syahirah, Mohd Faizal Abdollah, Zul Azri Muhamad Noh, Mohd Zaki Mas' ud, Siti Rahayu Selamat, Robiah Yusof, and Universiti Teknikal Malaysia Melaka. "Revealing the Criterion on Botnet Detection Technique." IJCSI International Journal of Computer Science Issues 10, no. 2, 2013, pp. 208-215.

[2] Silva, Sérgio SC, Rodrigo MP Silva, Raquel CG Pinto, and Ronaldo M. Salles. "Botnets: A survey." Computer Networks, 2012.

[3] Goebel, Jan, and Thorsten Holz. "Rishi: Identify bot contaminated hosts by irc nickname evaluation." In Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, pp. 8-8. 2007.

[4] Park, Younghee, and Douglas S. Reeves. "Identification of bot commands by run-time execution monitoring." In Computer Security Applications Conference, 2009. ACSAC'09. Annual, pp. 321-330. IEEE, 2009.

[5] Kolbitsch, Clemens, Paolo Milani Comparetti, Christopher Kruegel, Engin Kirda, Xiao-yong Zhou, and XiaoFeng Wang. "Effective and Efficient Malware Detection at the End Host." In USENIX Security Symposium, pp. 351-366. 2009.

[6] Shin, Seungwon, Zhaoyan Xu, and Guofei Gu. "EFFORT: Efficient and effective bot malware detection." In INFOCOM, 2012 Proceedings IEEE, pp. 2846-2850. IEEE, 2012.

[7] Zhao, David, Issa Traore, Bassam Sayed, Wei Lu, Sherif Saad, Ali Ghorbani, and Dan Garant. "Botnet Detection based on Traffic Behavior Analysis and Flow Intervals." Computers & Security (2013).

[8] Zeng, Yuanyuan. "On detection of current and next-generation botnets." PhD diss., The University of Michigan, 2012.

[9] Xu, Zhaoyan, Lingfeng Chen, Guofei Gu, and Christopher Kruegel. "PeerPress: utilizing enemies' P2P strength against them." In Proceedings of the 2012 ACM conference on Computer and communications security, pp. 581-592. ACM, 2012.

[10] VirusTotal. https://www.virustotal.com/. accessd July 2013.

[11] Lin, Hsuan-Tien, Chih-Jen Lin, and Ruby C. Weng. "A note on Platts probabilistic outputs for support vector machines." Machine learning 68, no. 3 (2007): 267-276.

[12] Elhadi, Ammar AE, Mohd A. Maarof, and Ahmed H. Osman. "Malware Detection Based on Hybrid Signature Behaviour Application Programming Interface Call Graph." American Journal of Applied Sciences 9, no. 3 (2012): 283.

[13] Automated Malware Analysis - Cuckoo Sandbox. http://www.cuckoosandbox.org/. accessed July 2013.

[14] Open Malware — Community Malicious code research and analysis. http://www.offensivecomputing.net/. accessed July 2013.

[15] LIBSVM – A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm/. accessed July 2013.

[16] Guo, Haoran, Jianmin Pang, Yichi Zhang, Feng Yue, and Rongcai Zhao. "Hero: A novel malware detection framework based on binary translation." In Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on, vol. 1, pp. 411-415. IEEE, 2010.

[17] Anubis: Analyzing Unknown Binaries. http://anubis.iseclab.org/. accessed July 2013.

[18] ::mwanalysis::CWSandbox::. https://mwanalysis.org/. accessed July 2013.

[19] Hsiao, Shun-Wen, Yi-Ning Chen, Yeali S. Sun, and Meng Chang Chen. "Combining Dynamic Passive Analysis and Active Fingerprinting for Effective Bot Malware Detection in Virtualized Environments." In Network and System Security, pp. 699-706. Springer Berlin Heidelberg, 2013.