27th Iranian Conference on Electrical Engineering (ICEE2019)

PODBot: A New Botnet Detection Method by Host and Network-Based Analysis

Somayeh Esmaeili
Department of Computer Engineering and IT
Amirkabir University of Technology
Tehran. Iran
esmaili@aut.ac.ir

Hamid Reza Shahriari
Department of Computer Engineering and IT
Amirkabir University of Technology
Tehran. Iran
shahriari@aut.ac.ir

Abstract— **The growing use of smart phones equipped with a variety of sensors makes them an ideal place for criminal activities and attacks such as bots. Unfortunately, the existing methods have made detection either by using network traffic or just using static analysis. In this paper, a method with a tool called PODBot has been introduced. The detection is done based on both application features and network traffic analysis. PODBot was evaluated on a set of botnets from well-known types and it could accurately detect over 87% in high risk and 96% in very high risk. In addition, in qualitative comparisons of similar tasks, due to the combination of the detection methods, it has features that improve the method relative to the previous methods.**

Keywords— android, botnet, padbot, static analysis, traffic analysis, smart phone, mobile Introduction

I. Introduction

Increasing use of smart phones equipped with variety of sensors and communication systems has made them an ideal target for bot attacks. In addition to the wide communicative facilities of the smartphone, the capabilities and information available on these devices is also another motive in targeting these systems by botnet attacks. According to Fortinet's 2018 report [1], Android variants ranking in the top five of their Weekly Threat Briefs several times, so Android malware are raising rapidly.

In this paper, a solution is proposed for detecting smartphone botnets using both host and network based metrics. Because it is not related to the type of communication channel, by detecting in the host, all the bots, including those communicated through SMS or Bluetooth channels are detected. If the communication channel is the Internet, traffic analysis and network flow can be fruitful in detecting with better precision, because static analysis has a high false positive rate.

In Section 2, the basic concepts in the field of Smartphone Botnets and their issues are discussed. In the third section, we will review the related work. In the fourth section, the proposed method of detection is introduced in the form of a tool called PODBOT, and in the next section we will evaluate this method and in the final section we will summarize and present conclusion.

II. Background

The botnet is a set of bots attached to a control and command channel waiting for the commands of bot master to perform their malicious activity. The vital component of the botnet is the command and control infrastructure, or C&C, that can be central, distributed or hybrid. Mobile botnet is a kind of malware that is installed on a mobile device and communicates with a network administrator through one or more control channels. Although mobile botnets and computer botnets are very similar and their architecture is similar, they also have differences in aspects, including Infrastructure, Connectivity, and Lucrativeness and Detection method.

To identify these types of botnets, we face some issues. If the detection is performed on the host, restrictions such as battery, memory, and processor restrictions renders the detection procedures requiring large resources, practically inefficient. On the other hand, the depth of the channel and the variety of communication channels make it harder to find a botnet than traditional botnets. [7].and organizational editing before formatting. Please note sections A-D below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

III. Related Work

In this section, we review the important studies done in this area. These works can be categorized into three areas: detection of computer botnets, detection of general mobile malwares, and detection of mobile botnets.

In the early studies in this field, the features of the botnet and their division are merely discussed [4-6]. Moreover, works such as [8-13] suggest methods for detecting botnets. In each of these cases, a factor has been used for detection. The classification of Figure 1 is the most comprehensive classification that covers most diagnostic methods [14].

Serious works in the second part began in 2009 with [15] and [16]. The first article was the first work conducted on Android for detection of malwares. The next study was carried out in the same year using the static analysis approach using the combination of license rules. [17] And [21] presented a static and dynamic combination method for detection which had a low percentage of accuracy. [18-19], [22] each presented new features in detection. [20] Has also worked on identifying malwares of iOS systems. In [31] authors uncover the relationships between the majority of the analyzed botnet families and offer an insight into each malicious infrastructure. Another paper on this issue was presented in 2014 with a static analysis approach [3]. It uses static features to detect on the mobile device itself.

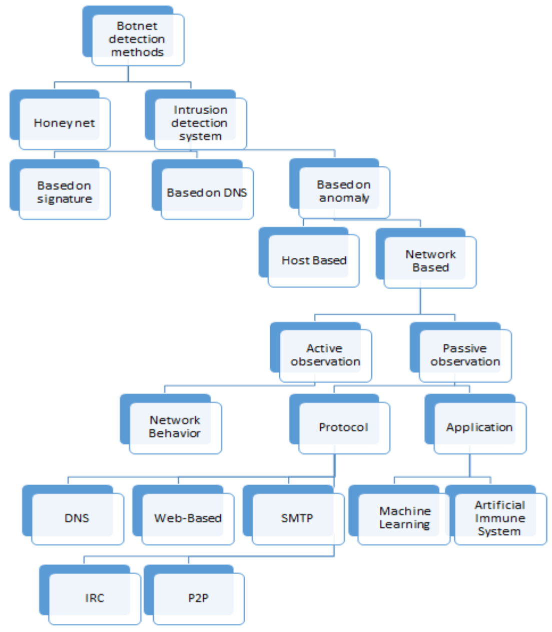

Fig. 1. Taxonomy of Botnet Detection Methods [3]

The works of the third part have generally been carried out since 2012. These works either have a static approach or have a dynamic analysis method. [7], have introduced a method to detecting mobile botnets. In [23], the detection of the push style mobile botnet is done, but [24] has found methods such as the previous article practically inefficient. Because of the use of non-static IPs and connectivity through different gateways to the Internet, push style method does not work in the mobile botnet. This paper presents a method for detecting pull style botnets by adding a white list and detection of signature that has been able to reduce the false positive rate. Another paper in this section is [25] which have been characterized by analyzing existing botnets. In [30], the inactive behavior analysis method has also been used to find bots in mobile. In the article [27], unlike previous work that examined measurements such as the number of packages and average, metrics such as studying periodicity, collaboration between nodes and similar behavior have been used.

IV. PODBOT DETECTION METHOD

The architecture of PODBot is indicated in Fig 2. We will continue to introduce the various sections of this tool, namely static analysis, traffic analysis and scoring system.

A. Static Analysis

In this stage, a dataset of APK files of botnet applications and normal applications were statically examined, and a set of features was extracted for each of them. These features are used to train a machine learning algorithm. After training, we use it as a classifier to sort the applications as a normal or botnet.

To select apps for learning, the Drebin Dataset was examined first [26]. Among these applications, those who did not run out of service, were selected, which were 70 apps. Subsequently, along with 52 normal applications, which were received from the Android markets such as Google play and the Café Bazar, static analysis was performed using Androguard, Androbug and MobSF tools, and their data was collected.

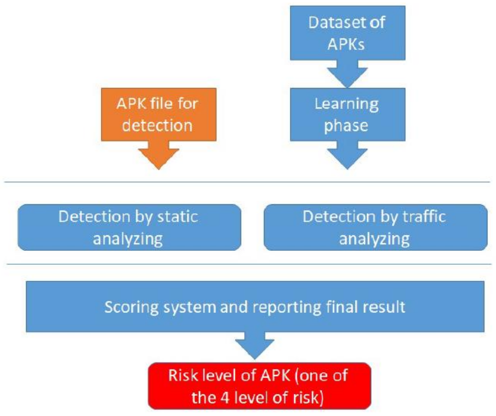
Dataset of APKs
APK file for detection
Learning phase
Detection by static analyzing
Detection by traffic analyzing
Scoring system and reporting final result
Risk level of APK (one of the 4 level of risk)

Fig. 2. Architecture of PODBOT

After this section, automatically, all the reports created by various tools were combined, the required items were extracted from them, and finally a general report of the static analysis of each app was obtained. According to this report, the amount of features -that we will be explained in the next section- was determined for each application. The next step was the cleaning of the data, which excluded features that did not affect the detection or drove the conclusion to the extremes. Using the remaining features and the allocation of each category, machine learning was carried out. By performing different experiments and a closer examination of each of these methods, the k-nearest neighbor method was chosen. In the detection phase, for each APK input file, a static analysis is initially performed, and after obtaining the static analysis report, the desired features were extracted from it. After the extraction step, the learning result with the attributes is given to the detection section based on the static analysis characteristics where the static detection is carried out. For better detection, we need features that differentiate between normal applications and botnets so that we can separate these two groups each from each other by extracting their relationships. By examining the methods in the previous work, we finally concluded that the properties selected in [3] can be used as the basis for static analysis.

However, this method has some problems, such as too many number of features in many classes and as a result, learning may not be accomplished well. , Also, there are some mutable features which may degrade the learning. Thus, in order proper use learning methods properly, it is needed to preprocess the features. After conducting several experiments and engineering features, we came up with some useful features. These features are: API calls, dangerous app permissions, intent, hardware features, content providers, and recipients of critical notifications and the number of dangerous URLs.

B. Traffic Analysis

Like the static analysis section, this section also consists of two phases of learning and detection. In the learning phase, first we installed 122 applications on a virtual machine to have static analysis on them. Then, by tpacketcapture pro tool, we listened to network traffic generated by each of them while doing various activities with the mobile and saved that for average of one hour.

After collecting traffic, we separated the HTTP and TLS packets and stored the required information from each one in the database. Some of these values are the source and destination IP, the source and destination ports, the package framing number, the packet sequence number and Ack number, the packet payload and so. After extracting the properties and collecting them for all samples, the result was given to the machine learning algorithm. Given that the number of these characteristics was less than that of static analysis, and since the number of samples was still limited, we applied the decision tree machine learning algorithm, because the tree was not spanned like the previous one, and it also used all the features.

In the detection phase, like learning, the APK file associated with the application is installed on the system and the traffic generated from it is recorded for 1 hour. Then the information from the HTTP and TLS packets is extracted and stored in the database. Then, the analysis of the characteristics of the database is performed, based on which the diagnosis is performed.

Most of the previous work related to the dynamic analysis of cybercriminals traffic, for example, [10-11], use statistical criteria such as flow, size, and time of packets. In study [27], which was presented in 2016, a set of criteria based on the calculation of the periodicity were used. None of these two types of features alone can perform a diagnosis well, because many botnets do not have the same statistical characteristics. For example, they don't have the same packet lengths. On the other hand, in order to prevent identification, some botnets also send their packets in arbitrary intervals. We introduced 3 feature categories to identify the types of botnets. The first category is the statistical characteristics, the second category is of the characteristics of the period, and the third category is of behavioral characteristics of the botnet.

The features used in this analysis are as follows:

- The average and standard deviation of the number of packets in time windows.
- The number of periodic packets in all time windows
- dispersion in time of periodic packets
- dispersion of the number of requests in each time window
- dispersion of packet payload size in time windows
- Ratio of request packets to response packets
- Sending confidential mobile data

C. *Scoring system and the result announcement system*

After conducting static and traffic analysis, the results of the decisions of the detection section, along with the features, are reported to the scoring system and the system of announcing the final result. In this section, according to the following rules, the final decision is made and the result that is one of the four the very high risk, high risk, medium risk, and low risk situations, is announced.

TABLE I.
SCORING RULES OF PODBOT

static analysis	traffic analysis	the result
botnet	botnet	very high risk to be botnet
botnet	normal	high risk to be botnet + feature vector
normal	botnet	average risk to be botnet + feature vector
normal	normal	low risk to be botnet – probably normal application

a.

These 4 levels make our detection stronger and more accurate. Further, we give the user chance to pick the granularity. The reason why we prioritized the result of static analysis to result of traffic analysis is that the traffic analysis feature vector does not design to detect all the botnets, but the static analysis has more capability to detect, so the probability that the detected botnet in static analysis will not be detected in traffic analysis is more and thus when static analysis detect the APK is botnet, the risk of being botnet in reality is higher than traffic analysis. Although we show feature vector to user in each of these situations to determine by himself/herself what features' value are more important for he/she in detection.

On the other hand, because security never is complete, we consider risk scenarios for all situations, since the result of the traffic analysis for botnets whose control channel is not the Internet and the botnets that are in the Incubation phase, is indicated as "Normal app ", and also the result of static analysis for botnets that do not have feature set, is "Normal app ", even if it send sensitive information from the phone.

V. EVALUATION OF PODBOT

Since the purpose of this project was to improve the previous methods, by evaluation this method in this section, we indicate that the PODBOT will have better performance and results than others.

A. *Evaluation of Static Analysis*

As mentioned in the previous section, according to the features selected for classification, as well as the number of botnet samples and normal applications, it is possible to use three algorithms: k-nearest neighbors, Naive Bayesian classifier, and the decision tree for learning. In TABLE II. , the evaluation results of each of these three algorithms are presented.

TABLE II. E
VALUATION OF STATIC DETECTION PHASE

Algorithm / Metrics	decision tree	KNN	Naïve Bayesian
precession	81.6%	75%	80.2%
Recall	88.6%	94.3%	92.9%
F-Measure	84.9%	83.5%	86.1%
False positive	11.4%	17.2%	13.1%
False negative	6.5%	3.2%	4%

By examining the values above, we conclude that if we want to consider the precision as the best metric for choosing the best method, the decision tree method has a higher precision.

B. *Evaluation of traffic Analysis*

In TABLE III. , like TABLE II. , different learning methods for traffic analysis have been evaluated.

TABLE III.
EVALUATION OF TRAFFIC ANALYSIS PHASE

Algorithm / Metrics	decision tree	KNN	Naïve Bayesian
precession	90.8%	90.5%	84.8%
Recall	84.3%	81.4%	80%
F-Measure	87.4%	85.7%	82.4%
False positive	4.9%	4.9%	8.1%
False negative	9%	10.6%	11.4%

By examining the values above, we conclude that the decision tree has a better percentage of precision and recall than the other methods, and its false positive and negative rates are lower. Therefore, the detection precision of this algorithm is more suitable for us.

C. *Total Evaluation*

As it shown in TABLE IV. There is no botnet that wrongly detected as a normal application by PODBot. And false negative, means normal apps that wrongly detected as botnet are just 2 sample that we think they are botnet, according to features of them.

TABLE IV. D
ETECTION RESULTS BASED ON NUMBER OF SAMPLES

	Botnets that wrongly detected as normal	Normal apps that wrongly detected as botnet
static analysis	4 samples	21 samples
traffic analysis	10 samples	6 samples
final result	-	2 samples

Only two normal apps in both units were mistakenly identified as botnets. The first is CoolReader, an eBook reader. This program uses an AD-server that has malicious URL. Its upload traffic also includes confidential mobile data (IMEI) and it seems to be a malicious app that exists on a trusted Android market. In its static analysis, there is also a permission to have access to dangerous API-calls such as access to device information along with Internet access permissions. The next sample is a Persian program called the Mizan. This sample also had its own dangerous URL server.

The final evaluation results have been indicated in **Error! Reference source not found.**

TABLE V. F
INAL RESULT OF PODBOT

Risk level / Metric	considering average risk and more	considering high risk and more	considering very high risk
precession	82.3%	87.1%	96.6%
Recall	100%	97.1%	81.4%
F-Measure	90.2%	91.8%	88.3%
False positive	17.6%	12.8%	3.3%
False negative	0%	2.3%	10.6%

According to these results, the user of PODBOT can consider his own desirable risk level according to the most important criterion in its own terms, but in normal circumstances, in order to have a moderate acceptable security, it can be considering apps with high and very high risk as Botnets.

D. *Comparing with other methods*

In order to show that our proposed method has improved the identification of the mobile botnets, we compare it with other papers designed to identify botnets and mobile malwares. The result of this comparison is shown in TABLE VI. As a result, PODBOT has some improvements like: detecting by both static and traffic analysis that improved the overall detection because it detects botnets having different C&C, not just Internet. Furthermore, it can be implemented in mobile devices as a standalone application. Moreover, another improvement in our detection method is “detection in several level”. With this feature, the user can use this method upon his request. After all we can say that PODBOT improved mobile botnet detection.

VI. SUMMARY AND CONCLUSION

In this article, we introduced the botnet and its various types. After that, the architecture of PODBot has presented and its main units such as static analysis, traffic analysis, and scoring were described. We evaluated the proposed method and found that in final result PODBot don’t detect any botnet wrongly as normal app. Also, it estimates average or high or very high risk for instants that were detected as botnet. We also compared the PODBOT method with those mentioned in the related works. Finally, we indicated the improvement our method achieved in the measurements mentioned in this study.

As a future work, we can add recovery after detection to this method, meaning that take mobile state to a safer state. Also we can implement a tool by dealing with CPU and memory limitation in learning phase for mobile devices.

TABLE VI. COMPARING PODBOT WITH OTHER METHODS

Feature / Method	Detection with static analysis	Detection with traffic analysis	Detection of botnets with different C&C	Focus on botnet detection	Detection in several level	Possibility of giving feature vector	Implement as a mobile app	Detection before installing APK	Recovery of mobile after detection
Choi[24]	-	✓	-	✓	-	-	-	-	-
Arp[3]	✓	-	✓	-	-	✓	✓	✓	-
Eslahi[27]	-	✓	✓	✓	✓	-	-	-	-
Nadji[22]	✓	✓	-	-	-	-	implementable	In runtime	✓
PODBOT	✓	✓	✓	✓	✓	✓	implementable	Half of detection	-

REFERENCES

[1] "Fortinet threat-report-q3--2018." [Online]. Available: https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/threat-report-q3-2018.pdf. [Accessed: 27-Jan-2019].

[2] E.Khoshhalpour, H.Shahriari, "BotRevealer: Behavioral Detection of Botnets based on BotnetLife-cycle," The Isc International Journal Of Information Security, Vol. 9, No. 2, PP. 1 - 7, 22 December 2017

[3] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket." in NDSS, 2014.

[4] Y. Zeng, "On detection of current and next-generation botnets," PhD Thesis, The University of Michigan, 2012.

[5] G. Gu, V. Yegneswaran, P. Porras, J. Stoll, W. Lee, "Active botnet probing to identify obscure command and control channels," in Annual Computer Security Applications Conference, 2009. ACSAC'09., 2009.

[6] A. Nappa, A. Fattori, M. Balduzzi, M. Dell'Amico, and L. Cavallaro, "Take a deep breath: a stealthy, resilient and cost-effective botnet using skype," in International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, 2010, pp. 81–100.

[7] A. Karim, S. A. A. Shah, and R. Salleh, "Mobile botnet attacks: a thematic taxonomy," in New Perspectives in Information Systems and Technologies, Volume 2, Springer, 2014, pp. 153–164.

[8] J.-S. Lee, H. Jeong, J.-H. Park, M. Kim, and B.-N. Noh, "The activity analysis of malicious http-based botnets using degree of periodic repeatability," in Security Technology, 2008. SECTECH'08. International Conference on, 2008, pp. 83–86.

[9] T.-M. Koo, H.-C. Chang, and G.-Q. Wei, "Construction P2P firewall HTTP-Botnet defense mechanism," in Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on, 2011, vol. 1, pp. 33–39.

[10] G. Gu, R. Perdisci, J. Zhang, W. Lee "BotMiner: Clustering Analysis of Network Traffic for Protocol-and Structure-Independent Botnet Detection.," in USENIX Security, 2008, vol. 5, pp. 139–154.

[11] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting botnet command and control channels in network traffic," 2008.

[12] W. Lu, M. Tavallaee, and A. A. Ghorbani, "Automatic discovery of botnet communities on large-scale communication networks," in Proceedings of the 4th International Symposium on Information, Computer, and Communications Security, 2009, pp. 1–10.

[13] B. Wang, Z. Li, D. Li, F. Liu, and H. Chen, "Modeling connections behavior for web-based bots detection," in 2010 2nd International Conference on E-business and Information System Security, 2010.

[14] A. Karim, R. B. Salleh, M. Shiraz, S. A. A. Shah, I. Awan, and N. B. Anuar, "Botnet detection techniques: review, future trends, and issues," J. Zhejiang Univ. Sci. C, vol. 15, no. 11, pp. 943–983, 2014.

[15] A.-D. Schmidt et al., "Static analysis of executables for collaborative malware detection on android," in 2009 IEEE International Conference on Communications, 2009, pp. 1–5.

[16] W. Enck, M. Ongtang, P. McDaniel, "On lightweight mobile phone application certification," in Proceedings of the 16th ACM CCS, 2009, pp. 235–245.

[17] T. Bläsing, L. Batyuk, A.-D. Schmidt, S. A. Camtepe, and S. Albayrak, "An android application sandbox system for suspicious software detection," in Malicious and unwanted software (MALWARE), 2010 5th international conference on, 2010, pp. 55–62.

[18] A. Shabtai and Y. Elovici, "Applying behavioral detection on android-based devices," in International Conference on Mobile Wireless Middleware, Operating Systems, and Applications, 2010, pp. 235–249.

[19] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for android," in Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, 2011, pp. 15–26.

[20] D. Damopoulos, G. Kambourakis, S. Gritzalis, and S. O. Park, "Exposing mobile malware from the inside (or what is your mobile app really doing?)," Peer--Peer Netw. Appl., vol. 7, no. 4, pp. 687–697, 2014.

[21] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang, "Hey, You, Get Off of My Market: Detecting Malicious Apps in Official and Alternative Android Markets." in NDSS, 2012, vol. 25, pp. 50–52.

[22] Y. Nadji, J. Giffin, and P. Traynor, "Automated remote repair for mobile malware," in Proceedings of the 27th Annual Computer Security Applications Conference, 2011, pp. 413–422.

[23] S. Zhao, P. P. Lee, J. Lui, X. Guan, X. Ma, and J. Tao, "Cloud-based push-styled mobile botnets: a case study of exploiting the cloud to device messaging service," in Proceedings of the 28th Annual Computer Security Applications Conference, 2012, pp. 119–128.

[24] B. Choi, S.-K. Choi, and K. Cho, "Detection of mobile botnet using VPN," in Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013, pp. 142–148.

[25] A. Karim, R. Salleh, M. K. Khan, A. Siddiqa, and K.-K. R. Choo, "On the analysis and detection of mobile botnet applications," J. Univers. Comput. Sci., vol. 22, no. 4, pp. 567–588, 2016.

[26] "The Drebin Dataset." [Online]. Available: https://www.sec.cs.tu-bs.de/~danarp/drebin/index.html. [Accessed: 16-Jan-2017].

[27] M. Eslahi, M. Yousefi, M. V. Naseri, Y. M. Yussof, N. M. Tahir, and H. Hashim, "Cooperative network behaviour analysis model for mobile Botnet detection," in Computer Applications & Industrial Electronics (ISCAIE), 2016 IEEE Symposium on, 2016, pp. 107–112.

[28] "Android Apps on Google Play." [Online]. Available: https://play.google.com/store. [Accessed: 16-Jan-2017].

[29] "Cafe Bazaar, Android Apps for Iranians." [Online]. Available: http://cafebazaar.ir/?l=en. [Accessed: 16-Jan-2017].

[30] M. Eslahi, R. Salleh, and N. B. Anuar, "MoBots: A new generation of botnets on mobile devices and networks," in IEEE Symposium on Computer Applications and Industrial Electronics (ISCAIE), 2012 , 2012, pp. 262–266

[31] Kadir, Andi Fitriah Abdul, Natalia Stakhanova, and Ali Akbar Ghorbani. "Android botnets: What urls are telling us." International Conference on Network and System Security. Springer, Cham, 2015.

[32] R. Aryan, H. Shahriari, "Botnet detection based on network behavioral and anomaly detection," 18th National Csi Computer Conference, Iran (Islamic Republic of), 12 March 2013 - 15 March 2014.