Malware Detection in Smartphones Using Network Traffic features

Vinisha Malik
PhD Scholar, Computer Science Engineering
MMU, Mullana, Ambala , India
Vinishamalik317@gmail.com

Dr Sandip Kumar Goyal
Associate Professor, Computer Science engineering
MMU, Mullana, Ambala , India
skgmmec@gmail.com

***Abstract*—**Smartphones have emerged as one of the ascendant computing platforms in today's era where Android has been the best cull for users as well as application developers due to its open source nature and feature affluent applications. Such popularity has invited an equipollent rise in malwares targeting Android. Since mobile contrivances sanction facile-to-use, touch-sensitive, and anywhere-anytime access to its resources, the mobile-categorical applications like SMS, MMS, Bluetooth, e-mail, and other accommodations may pose solemn threats and lead to financial losses and privacy leakages. Now a days researchers are trying to detect Android malwares. Very few researches are available considering network analysis in their detection techniques. In this paper, we are putting forward an algorithm to formulate network traffic traits with an ambition to curtail the number of traits considered for high malware detection accuracy.

***Keywords**-Network traffic; malware;android;*

*****

I. INTRODUCTION

Smartphones are becoming more puissant than early Per-sonal Computers (PCs) because of their fascinating com-puting capabilities. There has been a tremendous magnification in the sales of smartphones and is expected to reach around 2.6 billion by the year 2019 [1]. Google's Android has beaten all its competitors by an immensely colossal margin of proximately 80% [2]. At-tackers always target on impuissant systems and endeavor to assail on typically immensely colossal masses. Since 2010 Android has been the top-most target for malware developers as more than 90% of the mobile malware are targeted towards Android platform and the number is incrementing everyday. Techniques like repack-aging, update attacks and drive-by-downloads are utilized by assailers to inject malevolent payloads into mundane apps and app markets provide a facile channel for malignant samples to propagate to users. Threats posed by malware include financial loss, privacy leakage, root exploits and mobile bot-nets [3].The formatter will need to create these components, incorporating the applicable criteria that follow.

Albeit network traffic has been widely utilized in computer predicated intrusion detection but it has not been explored much in mobile malware detection. Network traffic features for mobile malware detection have been utilized by very few works as compared to the other features like sanctions, java code or system calls. Out of those studies, most of the works like [4] have utilized Android emulator for capturing network traffic of malware samples. The constraint with emulator predicated studies is that they do not fortify events like sending an SMS, dialing a number or rebooting of phone etc. Consequently malware that wait for such events may not trigger their maleficent payload and as a consequence may not communicate with the remote server, engendering no or very low amount of network traffic.

. Works like [13], [14] have focused on deviations in network traffic of malicious samples and normal samples. In this paper we aimed at analyzing network traffic comportment of Android malware by capturing their traffic from genuine smartphones, not the emulator and prioritized the traffic features so that a subset of features is utilized for detection rather than whole set of features. The benefits of such prioritization of features are: (a) reduction of features that may have, otherwise, negative impact on the detection performance; and (b) expeditious detection of maleficent network activity. Albeit traffic features have been analyzed in our precedent work [4] in which we found that seven network traffic features can distinguish between mundane traffic and malevolent traffic; but the traffic was captured from an emulator. As discussed earlier, there are circumscriptions in utilizing the emulator for capturing network traffic of Android malware. Consequently we used genuine smartphones to capture the network traffic.

The rest of the paper is organized as follows: Section 2 discusses the related work done in Android malware detection with focus on techniques using traffic features for analysis and detection. Proposed Methodology is presented in section 3. Results are discussed in section 4 and section 5 concludes the paper with future work directions.

II. LITERATURE SURVEY

Techniques considered for malware detection might be classified in two types: Static Analysis and Dynamic Analysis. Static Analysis handles verdict of dangerous permissions in the manifest file or the risky keywords within the source code of the application and this process is done without executing the application.

Enck et al [5] developed Kiran, one in all the earliest on-device defense resolution that gauged risks muddled with Android applications in terms of troubling permissions combination and provoked the set of security rules primarily based upon them. Droid-Analyzer [6], another static tool was designed to find root privileges among the Android malware. It distilled the list of risky keywords akin to malware samples and then hunted for these keywords in their testing information set of applications for banking etc. Permissions pattern formula [7] found high twenty permissions utilized by malicious further as traditional apps. They used each "requested per-missions" and "used permissions" to hunt the distinctive permission patterns of malware and traditional applications.

In [8] manifest file parts like intents, activities, services and broadcast receivers were thought of and bury component communication was analyzed between them. Machine learning algorithms like SVM, call Trees and Random Forests were used as detection model.

Measures like mutual data, T-Test and Correlation Coefficient were engaged in [9] to validate the highest permissions in malicious samples. Techniques like ordered

Forward choice and Principal element Analysis were accustomed determine risky permission sets. All such static solutions are cheap as their code or manifest file analyzed to relinquish the choice while not executing the applying. However such techniques lack in detecting lurking malware that transfer their malicious payload at run time. for instance malware that have update attack capability that transfer malicious element together with their updates, go unseen and so static solutions cannot sight such malware samples.

Dynamic Techniques are projected to beat restraints of static techniques during which run time suspicious behavior of applications is ascertained by executing the application on actual mobile device or on emulator. Traits like system calls and network traffic are engaged in observation run time behavior of applications. Researchers in [10] projected an on-device detection model to stop info leak from the mobile device. Information was tainted at four completely different levels: file level, technique level, message level and variable level. Burgera et al [11] possessed logs of system imply every application and applied clump to differentiate between traditional system calls and therefore the malicious ones. SCSDroid, a dynamic tool was proposed in [12]targeting repackaged applications from the third party applications.

III. METHODOLOGY

Real smartphones are utilized to capture their network behavior. A set of network traits is used in the work among which few traits will be distinguishing features. Distinguishing features here refer to the traits which have non overlapping values for malware and normal traffic. These traits are shown in Table I.

TABLE I. NETWORK TRAFFIC TRAITS AND THEIR RANGE VALUES

S. No.	***Traffic Trait Name (Notation)***
1.	Average Packet Size (T1)
2.	First Packet Size Sent (T2)
3.	First packet Size Received (T3)
4.	Ratio of Incoming to Outgoing bytes (T4)
5.	Bytes Sent per Flow (T5)
6.	Bytes Received per Flow (T6)
7.	Bytes Sent per Second (T7)
8.	Bytes Received per Second (T8)
9.	Packets Sent per Flow (T9)
10.	Packets Received per Flow (T10)
11.	Average Time interval between packets sent (T11)
12.	Average Time interval between packets received (T12)
13.	Ration of Incoming to Outgoing Packets (T13)
14.	Average Flow Duration (T14)
15.	Maximum Packet Size (T15)
16.	Minimum Time Interval between Packets Sent (T16)
17.	Maximum Time Interval between Packets Received (T17)

List of Android malwares that are utilized in work are proposed in Table II. These malwares are divided in two categories: one for the formulation or ranking of traits and another one is to test the effectiveness of the formulated set of traits.

TABLE II. LIST OF MALWARES

S. No.	**List of Malwares**		
	Malwares for formulation	***Malwares for Testing of Traits Set***	
1.	Plankton	DroidKungFu4	
2.	DroidKungFu1	DroidKungFu5	
3.	DroidKungFu2	GoldDream	
4.	DroidKungFu3	ADRD	
5.	DroidDream		
6.	DroidDreamLight		
7.			
8.			
9.			

A. *Trait Ranking Methodology*

Ranking the traits becomes a need here as it will lead to smaller set of traits so that testing time and running time will be less along with removal of irrelevant traits from the sets. This ranking involves two aspects namely: Information Gain and Chi-Square Test. Information Gain will give a measure of the reduction of uncertainty. Suppose S is the training set of traits $T_1, T_2\ldots..T_n$. C is the class label, IG represents the Information Gain which contributes to the ranking of traits. Higher the IG for any trait, higher will be the rank.

$$IG(CjT) = H(C)\ \ H(CjT) \qquad (1)$$

$$H(C)= {}_{i}{}^{X}P(C_i)\log_2(P(C_i)) \qquad (2)$$

$$H(CjT)={}^{X}P(T_i)\ {}^{X}P(C_jT_i)\log2(P(C_jTi)) \qquad (3)$$

Second aspect is the Chi-Square Test which measures the difference between expected and observed value and predicts whether the deviation is acceptable or not. This test is pertained separately on the normal traffic and malware traffic values. Lower value of deviation leads to the lesser value of Chi-Square test. Thus lower value of the test will lead the trait towards higher rank.

B. *Trait Assemblage*

Traits selection takes the path of following algorithm. We define two lists here: Initial List wth notation I_{list} which accommodates the common traits from all the three rankings i.e. one from IG and two from Chi-Square test, and O_{list} which embraces minimal set of traits. Detection results are measured in terms of $T_{measure.}$

$T_{measure}$ Score=2(Precision Recall)/(Precision + Recall)

Where Precision= $T_p/(T_p+F_p)$
Recall=$T_p/(T_n+F_n)$

Algorithm 1 Assortment of important traits from ranked traits

```
Input: Three rankings of features: $T_{list1}$, $T_{list2}$ and $T_{list3}$
obtained from I.G.,CHI on malware and CHI on normal
respectively
Output: Minimal set of features which gives best detection
results
1: $I_{list}$ ; , $O_{list}$ ; , $F_{max} = 0$
2: for k = 1 ! 22 Do
    3:temp $(T_{list1}))_k \ (T_{list2})_k \ (T_{list3})_k$
        4: $I_{list} = I_{list}$ [ ($I_{list}$ { temp)
        5: Run Na•ve Bayes' classi er on test data using traits in $I_{list}$
        6: Calculate $T_{measure}$ score $T_{score}$ of detection results
        7: if $T_{score} > TF_{max}$ then
            8: $F_{max}$ = $F_{score}$
            9: $O_{list}$ =$I_{list}$
        10: end if
11: end for
12: Return $O_{list}$
```

IV. CONCLUSION

We have projected the technique to rank the traffic traits and then to choose the minimal set among all the available traits. This will reduce the testing and execution time as irrelevant traits will be thrown out of consideration.

REFERENCES

[1] Statista. Number of smartphone users worldwide from 2014 to 2019 (in millions). http://www.statista.com/statistics/330695/ number-of-smartphone-users-worldwide/, 2016. [Online; accessed 01-March-2016].

[2] Smartphone OS Market Share, 2015 Q2. http://www. idc.com/prodserv/smartphone-os-market-share.jsp, 2015. [Online; accessed 01-March-2016].

[3] Y. Zhou and X. Jiang. Dissecting android malware: Characterization and evolution. In 2012 IEEE Symposium on Security and Privacy (SP), pages 95{109, May 2012.

[4] A. Arora, S. Garg and S. K. Peddoju. Malware detection using network tra c analysis in android based mobile devices. In 8th International Conference on Next Generation Mobile Apps, Services and Technologies (NGMAST), pages 66{71, Sept 2014.

[5] W. Enck, M. Ongtang and P. McDaniel. On lightweight mobile phone application certi cation. In16th ACM conference on Computer and Communications Security, pages 235{245, 2009.

[6] S.H. Seo, A. Gupta, A. M. Sallam, E. Bertino, and K. Yim. Detecting mobile malware threats to homeland security through static analysis. Journal of Network and Computer Applications, 38:43 { 53, 2014.

[7] V. Moonsamy, J. Rong, and S. Liu. Mining permission patterns for contrasting clean and malicious android applications. Future Generation Computer Systems, 36:122 { 132, 2014.

[8] K. Xu, Y. Li and H.R Deng. ICC Detector: ICC-Based Malware Detection on Android. IEEE Transactions on Information Forensics and Security, 11:1252-1264, 2016.

[9] W. Wang, X. Wang, D. Feng, J. Liu, Z. Han and X Zhang. Exploring permission-induced risk in Android applications for malicious application detection. IEEE Transactions on Information Forensics and Security, 9:1869-1882, 2014.

[10] E. William, G. Peter, C. Byunggon and C. Landon. TaintDroid: An information •nCow tracking system for real time privacy monitoring on smartphones. Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation, 2011.

[11] I. Burgera, U. Zurutuza and S. Nadjm-Tehrani. Crowdroid: Behavior-Based Malware Detection System for Android. Proceedings of the 1st Workshop on Security and Privacy in Smartphones and Mobile Devices, 2011.

[12] Y. Lin, Y.C. Lai, C.H. Chen, and H.C. Tsai. Identifying android malicious repackaged applications by thread-grained system call sequences. Computers and Security, 39, Part B:340 { 350, 2013.

[13] A. Shabtai, L. Tenenboim-Chekina, D. Mimran, L. Rokach, B. Shapira, and Y. Elovici. Mobile malware detection through analysis of deviations in application network behavior. Computers and Security, 43:1 { 18, 2014.

[14] Z. Chen, H. Han, Q. Yan, B. Yang, L. Peng, L. Zhang, and J. Li. A rst look at android malware traffic in first few minutes. In 2015 IEEE Trustcom/Big DataSE/ISPA, pages 206{213, Aug 2015.