Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

Information Flow based Malware Detection System for Android

S. Kalaiarasi, Thota Srikiran, Vulla Ravi Teja, Konathala Vamsi Krishna,
Bathina Ramachandra Rao
Computer Science and Engineering,
SRM Institute of Science and Technology, Chennai, India.

***Abstract*: In this paper we propose another method to distinguish the malware in android gadgets utilizing data stream based examination and profound learning algorithms. In this methodology the framework analyzes the structure of data streams. Data stream is the development of data among individuals and a PC. Proficient and secure data streams are a focal factor in the execution of basic leadership, procedures and communication. The structure of the stream contains designs which are utilized to distinguish the conduct present in them. A procedure called the N-gram examination which is utilized to recognize the examples in complex flows. The N-gram investigation is performed on groupings of API calls that happen along Complex-Flows' control stream ways. We show accuracy by testing the framework on different applications.**

Index Terms*: *Deep learning, Information Flows, N-Gram. Analysis, API calls, SVM Algorithm

I. INTRODUCTION

Lately as the innovation is expanding step by step it is critical to verify our information and touchy data and keep them secret from others. But since of expanded innovation it wound up less demanding to hack into somebody's information and stole the information from them causing so much misfortune. Android gadgets have turned into a piece of our day by day lives. It is normal that by 2019 the quantity of clients utilizing an Android gadget will achieve five million. So the Android has turned into an essential wellspring of focus for the programmers to hack the information. They use malware to contaminate the gadget and transmit the client information in the gadget to the programmer. A malware is a vindictive code which shrouds itself in a considerate application and runs complex calculations once it is contaminated.

There are numerous past frameworks that are available to recognize the malware in the Android gadget. Albeit a few frameworks are effective, the counter malware creators have created different methods against the malware. In these past methodologies the streams are absent so the engineers utilized the battery utilization as a factor to recognize the malware. However at this point the data streams are available in the applications which are utilized to distinguish the streams and examine their structure and conduct to recognize the malware. A few calculations are likewise used to help recognize the malware in the kindhearted applications. The distinction between the pernicious application and the kind application is the manner by which the data stream are created and application conduct amid calculation of delicate data. A considerate application processes the touchy data and the after that the calculation stops yet if there should arise an occurrence of pernicious application it keeps on doing the calculations and duplicating or sending the utilization information to the programmers. To recognize these practices of the malignant applications the perplexing streams are utilized. A Complex-Flow is a great deal of clear (source, sink) streams that share a run of the mill bit of code in a program. For example, a program can scrutinize contact information, scramble it, store it, and send it over the Internet. The objective of this paper is to discover the distinction by dissecting the conduct and examples by utilizing complex streams and profound learning strategies by utilizing calculations called as SVM algorithm and N-gram analysis.

II. SYSTEM DESIGN

2.1 Proposed System

In this undertaking we demonstrate that there is have to look past the typical straightforward streams so as to distinguish the vindictive applications and malware present in it. The malware have developed and rather than essentially gathering the information and quickly uncovering it, the malware is playing out the mind boggling computations and furthermore changing the application conduct. So as to identify the malware we utilize the dynamic investigation procedure to distinguish the malware.Our proposed system uses the following features:

- String feature
- Method opcode feature
- Method API feature
- Shared library function opcode feature
- Permission feature
- Component feature
- Environmental feature

2.1.1 Proposed System Advantages

- It can mirror the attributes of Android applications.
- It can speak to malware qualities adequately notwithstanding when malware shares numerous regular properties with kind applications.
- High detection exactness
- It upgrade the general exactness of the model
- It decrease the preparation time..

2.2 Multi Flows

The goal of the Multi-Flow detection algorithm is to:

1 Generate a global graph of complete data stream paths for an application, and

2 Identify the convergence between individual data stream ways that speak to Multi-Flows. Here, the crossing point of two

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

data stream ways basically implies two data stream ways share no less than one hub in the worldwide diagram.

3 These multi streams are utilized to identify the application practices utilizing the examples. We utilized these streams to test on some generous applications to take note of the pernicious practices of the application.

III. SYSTEM ARCHITECTURE

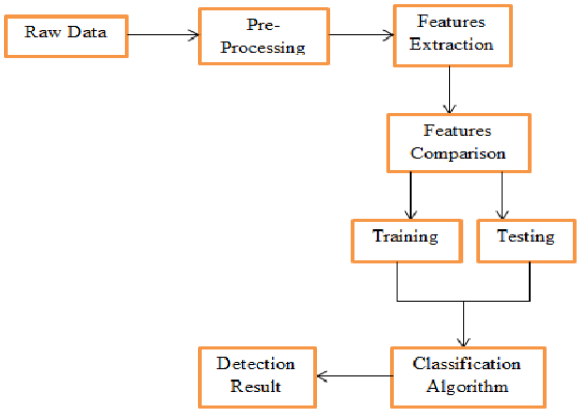

Fig. 1 System Architecure

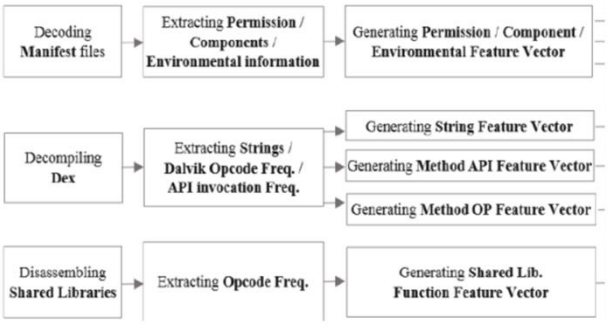

Fig. 2 Feature Extraction

As appeared in the Fig 1 the crude information is transfered into a compiler where it is handled and the highlights are extricated as appeared in the Fig 2. After the highlights are separated these highlights are contrasted and the current highlights. At that point the preparation is given to the engineers dependent on the highlights and testing s done. The incorporated code is sent into a characterization calculation which is the SVM joined with the N-Gram investigation to distinguish the consequences of the code and locate the vindictive pieces of code in the application information.

The element extraction process is the fundamental procedure where the application bundle is part into their comparing records like the show documents, Dex documents, and Shared Libraries. After the extraction procedure each record is additionally extricated to make it increasingly executable for the processor. The show documents are extricated of creating Permissions, Components and Environmental Feature Vector. The Dex documents are decompiles and sent to Dalvik Virtual Machine which creates the Dalvik Opcode frequencies and API Invocation frequencies. These are additionally used to produce the Vectors as appeared in Fig. 2. The mutual libraries are dismantled to create Opcode frequencies and this is utilized to separate the vector as appeared in the Fig.2.

IV. WORK FLOW

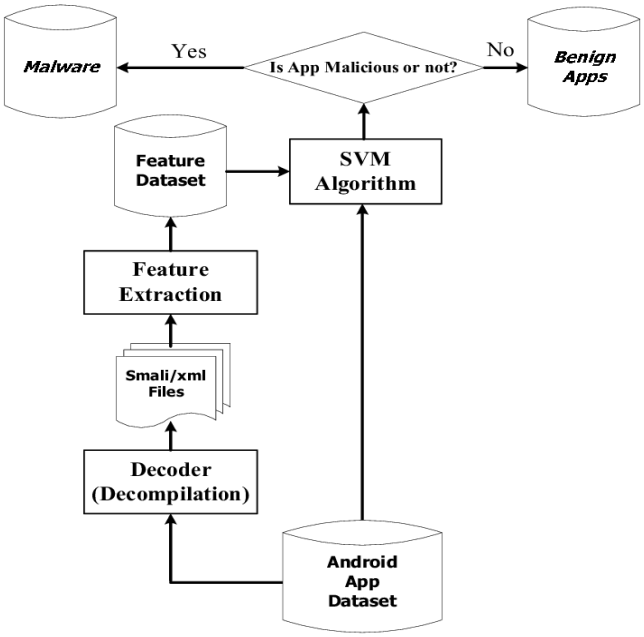

Fig. 3 Work Flow Graph

4.1 SVM Algorithm

"Bolster Vector Machine" (SVM) is a managed AI computation which can be used for both portrayal or backslide challenges, it is generally used in course of action issues. In this computation, we plot each datum thing as a point in n-dimensional space (where n is number of features you have) with the estimation of every segment being the estimation of a particular encourage. By then, we perform gathering by finding the hyper-plane that different the two classes extraordinary .

4.2 N-Gram Algorithm

The fundamental purpose of n-grams is that they catch the language structure from the factual perspective, similar to what letter or word is probably going to pursue the given one. The more drawn out the n-gram (the higher the n), the more setting you need to work with. Ideal length truly relies upon the application. On the off chance that your n-grams are excessively short, you may neglect to catch imperative contrasts. Then again, on the off chance that they are excessively long, you may neglect to catch the "general knowledge" and only stick to particular cases.

V. MODULES

The modules are classified into three parts

1 Raw Data Extraction Process

2 Feature Extraction Process

3 Feature Vector Generation Process

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

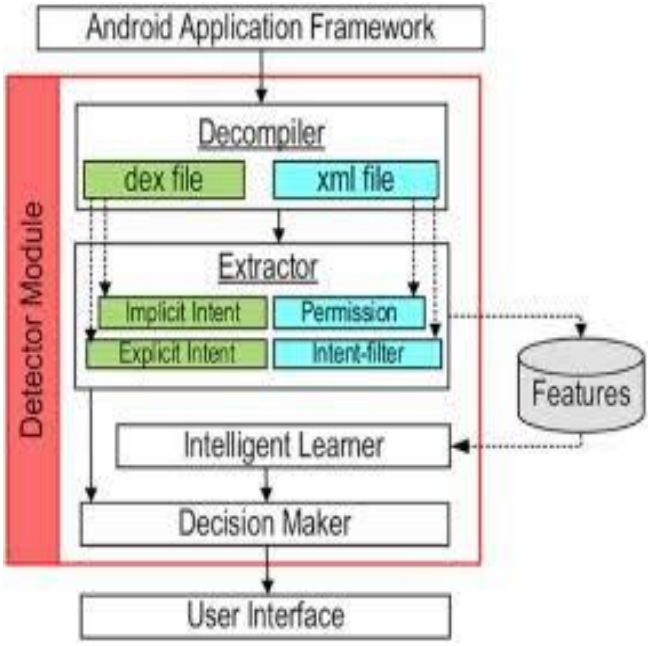

Fig.3 Module Description

5.1.1 Raw Data Extraction Process

This extraction process is performed to make the apk records compilable. To separate the crude information, an Android package document is unfastened, and a show record, a DEX document, and distributed library records are extricated first. The show document and the dex record are decoded or dismantled by APKtool, and the common library records (for example .so documents) in the bundle can be dismantled.

5.1.2 Feature Extraction Process

This process is directed to acquire the basic component information from the crude information. To begin with, strategy opcode highlights and technique API highlights are removed from little documents which have the dismantled consequences of the dex record. The little record is isolated into the technique squares, and, by examining Dalvik bytecodes, the Dalvik opcode recurrence of every strategy is determined. Likewise, amid the bytecode examining, it is checked whether the conjuring of the hazardous APIs exists in the strategy, and the unsafe API summon recurrence of every technique is determined.

5.1.3 Feature Vector Generation Process

The removed features includes in the past procedure are utilized to form highlight vectors. Seven sorts of the component vector are produced from separated highlights. The seven element vectors are partitioned into two sorts as indicated by their element portrayals: presence based component vectors and likeness based element vectors. The presence based element vector is the element vector whose components just speak to the presence of highlights in the vindictive element database, and instances of these are string, consent, part and natural element vectors.

VI. EVALUATION

6.1 Methodology and Metrics

We utilized four arrangements of various mixes in our trials to assess the order framework.

The evaluation process is as follows:

- In this we utilized a ten times cross approval system to partition the applications into sets called preparing and testing sets. The classifiers is prepared on highlight vectors to frame arbitrary 90 percent of both considerate and pernicious applications. The grouping procedure will be rehashed a few times altogether and the normal is determined.
- The preparing set depends on both good and malignant applications. The N-grams that are produced from these applications are utilized to frame global element space. A component vector is assembled dependent on N-grams.
- After preparing the classifiers we utilize the testing set of blended applications for the order. The classifiers at that point gives the choice on an application, in view of the N-grams highlight vector, as either 'malicious' or 'good'.

The following

TP **"**True positive rate"—the rate at which a good app is detected as a good app

TN **"**True negative rate"—the rate at which a malware is detected perfectly as malware app

FP " False positive rate"—the rate at which the malware is detected as a good app incorrectly.

FN "False negative rate"—the rate at which the good app is detected as a malware

VII. RESULT

7.1 Google Play Apps vs Malware Apps

In this area, we structured distinctive analyses to assess our framework completely dependent on favorable applications and present day malware applications. In the first place, we look at the previous and recent Google Play applications against present day pernicious applications exclusively; at that point, we run examination on all amiable applications and noxious applications. To do this, we isolate present day pernicious applications arbitrarily into two sets to coordinate with previous and recent generous applications as needs be and name them as Malware_one and Malware_two. The point by point results are talked about underneath.

Table 1
Classification Results on Play_2018 vs Malware_2 Apps

gram size	TP	TN	FP	FN	accuracy
1	0.921	0.767	0.233	0.079	0.838
2	0.841	0.863	0.137	0.159	0.853
3	0.619	0.863	0.137	0.381	0.75
4	0.475	0.948	0.052	0.525	0.743
5	0.424	0.948	0.052	0.576	0.721
1,2	0.968	0.849	0.151	0.032	0.904
1,2,3	0.857	0.877	0.123	0.143	0.868
1,2,3,4	0.683	0.877	0.123	0.317	0.787
1,2,3,4,5	0.667	0.89	0.11	0.333	0.787

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

7.2 2018 Playstore apps vs Modern Malware Apps

We assess our methodology on various arrangements of applications. In this we have utilized the latest Google play applications, marked as Play_2018, for our amiable applications set. We have picked an alternate arrangement of vindictive applications, which are named as Malware_2. The outcomes are appeared in the Table 1.The outcomes demonstrate a comparable conduct as we increment the gram size.by utilizing this we can accomplish exceptionally exact characterization of the applications, while using the false positive rates minimum. Littler gram values gives us the better precision for both kind applications and malevolent applications.

Table 2

Classification Results on Play_2018 vs Malware_1 Apps

gram size	TP	TN	FP	FN	accuracy
1	0.937	0.795	0.205	0.063	0.86
2	0.937	0.781	0.219	0.063	0.853
3	0.841	0.904	0.096	0.159	0.875
4	0.441	0.883	0.117	0.559	0.691
5	0.39	0.909	0.091	0.61	0.684
1,2	0.937	0.836	0.164	0.063	0.882
1,2,3	0.825	0.89	0.11	0.175	0.86
1,2,3,4	0.703	0.909	0.091	0.297	0.849
1,2,3,4,5	0.688	0.909	0.091	0.313	0.844

Moreover we additionally ran the order on the latest playstore applications vs alternate arrangements of vindictive apps due to malware, The assessment results have appeared in the above table.As we can see that outcomes are familiar. So as should be obvious that the cutting edge malware applications are increasingly like kindhearted applications as indicated by the utilization of the single APIs.

Appset	TP	TN	FP	FN	Accuracy
Play_2016vsMalware_1	0.87	0.62	0.39	0.14	0.73
Play_2018vsMalware_2	0.59	0.82	0.18	0.42	0.72

VIII. DISCUSSION

Data streams themselves may not give enough data to recognize malware applications. Point by point application conduct, can be caught by utilizing the N-grams, it is a vital element that can give the basic data used to recognize noxious applications from kindhearted applications. The point by point application conduct gathered by Information-Flow gives more proof of the noxiousness of an application. For instance, assume an accompanying perception recognized by the examination. Comparable, long API call grouping are less normal crosswise over amiable applications, showing that kind applications differ incredibly in application conduct. In any case, long API call grouping are normal crosswise over malware applications and can enhance the identification rate of malevolent applications, demonstrating malware shares similar standards of conduct. Diverse values of the N-grams indicates toughness of the application conduct. Numerous MalGenome applications can be characterized independently from kindhearted applications dependent on gram-1 includes alone, which means these applications show critical contrast of application conduct on single API versus benevolent applications. Interestingly, grouping of other current malware applications has to meet requirements more than gram-1 highlight.This considers these type of malwares are most similar with the other application than the MalGenome. Be that as it may, they can in any case be separated from typical applications by dissecting itemized application practices spoken to by various gram highlights.

IX. CONCLUSION

We hereby proposed a new idea of Information Flows to infer application conduct on gadget touchy information. We likewise present a robotized arrangement framework that influences application conduct alongside application data streams for grouping kind and noxious Android applications. We have point by point our way to deal with find Complex Flows in an application, separate application conduct includes, and apply an arrangement strategy. We demonstrate the viability of our grouping framework by exhibiting assessment resulting in the Play Store applications and some vindictive applications. The future work is to plan the filtering N-Grams which includes the extraction of disposing the non-effective edge work of API calls We additionally can use other AI grouping methods to locate the best ones.

REFERENCES

[1] Y. Aafer, W. Du, and H. Yin. Droidapiminer: Mining api- level features for robust malware detection in android. In Proc. of SecureComm 2013, 2013.W.-K. Chen, Linear Networks and Systems (Book style).Belmont, CA: Wadsworth, 1993, pp. 123–135.

[2] V. M. Afonso, P. L. de Geus, A. Bianchi, Y. Fratantonio, C. Kruegel, G. Vigna, A. Doupe´, and M. Polino. Going native: Using a large-scale analysis of android apps to create a practical native-code sandboxing policy. In Proc. of NDSS 2016, 2016.

[3] D. Arp, M. Spreitzenbarth, H. Gascon, and K. Rieck.Drebin: Effective and explainable detection of android malware in your pocket, 2014.

[4] S. Arzt and E. Bodden. Stubdroid: Automatic inference of precise data-flow summaries for the android framework. In Proc. of ICSE 16, 2016.

[5] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. L. Traon, D. Octeau, and P. McDaniel. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle- aware taint analysis for android appstion in tcb source code. In PLDI '14, Edinburgh, UK, 2014.

[6] V. Avdiienko, K. Kuznetsov, A. Gorla, A. Zeller, S. Arzt, S. Rasthofer, and E. Bodden. Mining apps for abnormal usage of sensitive data. In ICSE '15, Piscataway, NJ, USA,2015.

[7] P. Barros, R. Just, S. Millstein, P. Vines, W. Dietl, M. d'Amorim, and M. D. Ernst. Static analysis of implicit control flow: Resolving Java reflection and Android intents. In ASE '15, Lincoln, NE, USA, 2015.

[8] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani. Crowdroid: behavior-based malware detection system for android. In SPSM '11, 2011.