Survey Paper is on Android Malware Detection Using Deep Eigenspace Learning

Mr. Rahul Pawar, Akanksha Khedkar, Golu Kumar, Apurva Dhumal
Department of Computer Engineering,
Dr. D. Y. Patil College of Engineering, Akurdi, Pune, India.

ABSTRACT

Malwares are spreading round the world and infecting not solely the top users however additionally large organizations and service providers. Android operating system looks to own attracted the foremost attention from malicious code writer thanks to its popularity. Earlier, Signature based detection techniques were won’t to observe unknown malwares. But it absolutely was lean because these techniques were not able to observe unknown malwares (0-day attack). To analyse the malwares, static and dynamic techniques are used. Static analysis has advantage of being undetectable, as malware cannot modify its behaviour throughout analysis. Despite range of detections and analysis techniques are in situ, high detection accuracy of recent malwares area unit still an important issue. This survey paper highlights the prevailing detection and existing analysis strategies used for the android malicious code.

I. INTRODUCTION

In the past, the installation of applications was a source of issues for the users as a result of there wasn’t a centralized web site to transfer their applications and that the wont to search them in the web. Many operational systems like Symbian, in an effort to avoid piracy and defend the device, used an authentication protocol that certified apk or application and, usually, caused several inconveniences to the users. Nowadays, new ways for distribution and installation have been developed because of the wide used web affiliation gift within the mobile devices. Therefore, users will install any application they require, avoiding the affiliation of the device to a private pc. The App Store of Apple was the first on-line store to bring this new paradigm to novel users. The model was praised and it became terribly productive, leading to alternative vendors like RIM, Microsoft or Google to adopt constant business model and developing application stores for his or her devices. These factors have led an oversized variety of developers to specialize in these platforms.

Android may be a free, open supply OS designed on UNIX operating system for mobile devices, together with the UNIX kernel, the Intermediate layer, the application Framework layer and also the Application layer. The UNIX kernel provides basic system services, like process scheduling, C.P.U. management. Intermediate layer embrace humanoid native libraries and humanoid runtime atmosphere. Android permission mechanism is an access management mechanism. Associate degree application has no ability to perform the corresponding operation unless it's requested the proper permission. Android provides concerning one hundred thirty permissions for system resources protection, and provides a collection of corresponding API to access to system resources.
Android supports a booming third party application market. As of July 2011, the android Market enclosed quite 250,000 applications that are downloaded quite six billion times. Sadly, the expansion within the android platform has triggered the interest of unscrupulous application developers. Humanoid grayware collects excessive amounts of personal info (e.g., for aggressive promoting campaigns), and malware harvests information or sends premium SMS messages for profit. Grayware and malware have each been found in the humanoid Market, and also the rate of latest malware is increasing over time Google doesn't review or limit humanoid applications. Instead, humanoid uses permissions to alert users to privacy- or security invasive applications. Once a user initiates the method of installing associate degree application, he or she is shown the list of permissions that the appliance requests. This list identifies all of the phone resources that the appliance can have access to if it's put in. for instance, an application with the SEND SMS permission will send text messages, however an application while not that permission cannot. If the user isn't comfy with the applications permission requests, then he or she will cancel the installation. Users are not shown per- missions at any time aside from installation.

II. LITERATURE SURVEY

III. 1. We have referred “SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System” this paper.

Methodology:
In this paper, a novel 3-level hybrid Android malware detection model is proposed named as SAMADroid. It is a hybrid between three levels for malware analysis and detection:

i) Static & Dynamic Analysis
ii) Local & Remote Host
iii) Machine Learning Intelligence.

Result:
Two experiments that are performed to determine the performance overhead caused by SAMADroid on real Android device. Firstly, the performance of Android device is observed through Benchmark tool before and after running SAMADroid. This experiment provides the overhead caused by the SAMADroid client application while running on the real Android device. Secondly, it provides the performance overhead of SAMADroid with MADAM. Results shows that performance overhead caused by MADAM is high in comparison to SAMADroid.

Future Scope:
Although SAMADroid provides high detection accuracy at low resource consumption but it has some limitations too. The whole system is dependent on server communication. Thus, if the network link gets down or congestion occurs at channel due to which Android device cannot communicate with the server then the performance of SAMADroid will be reduced. So Future scope for this methodology is to enhance the malware dataset for SAMADroid, including the recent malwares so that SAMADroid effectively secures the Android applications against recent malware applications.

2. We have referred “A Combination Method for Android Malware Detection Based on Control Flow Graphs and Machine Learning Algorithms” this paper.

Methodology:
In this paper, they present a machine learning method for Android malware detection which can automatically detect known and unknown types of malware if it
belongs to the malware types that they have analyzed. In this methods, they first decompile the Android application and construct the control flow graph (CFG) from the source code. Then they extract Application Program Interface (API) calls from the CFG and build three different types of API data sets: Boolean data sets, frequency data sets and chronological data sets. Build 3 detection models: API Usage Detection Model, API Frequency Detection Model and API Sequence Detection Model based on the data sets using machine learning methods.

Result:
The experiments were conducted on 10010 benign applications and 10683 malicious applications. The results show that detection model achieves 98.98% detection precision and has high accuracy and stability. They evaluate the accuracy of each model using standard classification metrics - Precision, Recall and F-score - and compare the performance of these three model. With the combination method, an ensemble model is constructed and achieves 98.98% detection Precision.

Future Scope:
Detecting the malicious families that malwares belong to. In this paper, the models they build are 2-class classification models, which means they can only determine whether the application is malicious or not. In the future, will build a multi-class classification model to determine which malicious family the application belong to if detected malicious

1. We have referred “Detection of Malicious Code Variants Based on Deep Learning” this paper.

Methodology:
This paper proposed a novel method that used deep learning to improve the detection of malware variants. Deep learning demonstrated excellent performance in image recognition. To implement our proposed detection method, they converted the malicious code into grayscale images. Then, the images were identified and classified using a convolutional neural network (CNN) that could extract the features of the malware images automatically. In addition, they utilized a bat algorithm to address the data imbalance among different malware families.

Result:
To validate the effectiveness and efficiency of the proposed approach, they designed experiments to
1) Verify the efficiency of different data equalization methods
2) Ascertain the effects of different malware image sizes, and
3) Compare the results of proposed approach with the outcomes of other methods for detecting malicious code.

The results demonstrated that this model achieved good accuracy and speed as compared with other malware detection models.

Future Scope:
In future work, SPP-net model will developed to allow images of any size to be used as input. The SPP-net can extract features at variable scales. They can introduce that layer into model between the last subsampling layer and the fully connected layer to improve models flexibility. In addition, the transformation of malicious code into colour images would be a good topic for future research.

1. We have referred “Significant Permission Identification for Machine-Learning-Based Android Malware Detection” this paper

Methodology:
In this paper, they introduce Significant Permission Identification (SigPID), a malware detection system based on permission usage analysis to cope with the
rapid increase in the number of Android malware. Instead of extracting and analysing all Android permissions, they develop three levels of pruning by mining the permission data to identify the most significant permissions that can be effective in distinguishing between benign and malicious apps.

Result:

The results indicate that when a support vector machine is used as the classifier, can achieve over 90% of precision, recall, accuracy, and F-measure, which are about the same as those produced by the baseline approach while incurring the analysis times that are 4–32 times less than those of using all permissions. Compared against other state-of-the-art approaches, SigPID is more effective by detecting 93.62% of malware in the dataset and 91.4% unknown/new malware samples.

Future Scope:

The permission lists used by DREBIN contain many meaningless features. It is possible that performance improvements can be achieved by integrating SigPID with FT into DREBIN to improve both malware detection accuracy and running time performance. So this can be a future scope.

1. We have referred “Robust Malware Detection for Internet of (Battlefield) Things Devices Using Deep Eigenspace Learning”

Methodology:

In this paper, they have present a deep learning based method to detect Internet of Battlefield Things (IoBT) malware via the device’s Operational Code (OpCode) sequence. Firstly they transmute OpCodes into a vector space and apply a deep Eigenspace learning approach to classify malicious and benign applications. They also demonstrate the robustness of our proposed approach in malware detection and its sustainability against junk code insertion attacks.

Result: A graph of selected features was created for each sample and a deep Eigenspace learning approach was used for malware classification. Paper evaluations demonstrated the robustness of our approach in malware detection with an accuracy rate of 98.37 percent and a precision rate of 98.59 percent, as well as the capability to mitigate junk code insertion attacks.

Future Scope:

Author planned to evaluate our approach against larger and more diverse datasets, as well as implementing a prototype of the proposed approach in a real-world IoT and IoBT system for evaluation and refinement.

1. We have referred “A Multimodal Deep Learning Method for Android Malware Detection Using Various Features” this paper

Methodology:

This paper proposes a novel framework for android malware detection. Framework uses various kinds of features to reflect the properties of android applications from various aspects, and the features are refined using our existence-based or similarity-based feature extraction method for effective feature representation on malware detection. A multimodal deep learning method is proposed to be used as a malware detection model.

Result:

Author confirms the usefulness of the feature and their proposed feature vector generation method. And authors also carried out experiments about the applicability on the unsupervised learning based classification and the obfuscation resilience. As a result, our framework was effective enough to be used in the Android malware detection.
Future Scope:

Author proposed Android malware detection framework that utilizes many static features to reflect the properties of applications in various aspects not dynamically generate features.

<table>
<thead>
<tr>
<th>Sr.No.</th>
<th>Title</th>
<th>Year</th>
<th>Author</th>
<th>Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System</td>
<td>January 2018</td>
<td>SABA ARSHAD, MUNAM A.SHAH, ABDUL WAHID</td>
<td>Novel 3-level hybrid Android malware detection model is proposed named as SAMADroid</td>
</tr>
<tr>
<td>3.</td>
<td>Detection of Malicious Code Variants Based on Deep Learning</td>
<td>July 2018</td>
<td>Zhihua Cui, Fei Xue, Xingjuan Cai</td>
<td>Used deep learning to improve the detection of malware variants.</td>
</tr>
<tr>
<td>4.</td>
<td>Significant Permission Identification for Machine-Learning-Based Android Malware Detection</td>
<td>July 2018</td>
<td>Jin Li, Lichao Sun, Qiben Yan</td>
<td>Used Significant Permission IDentification (SigPID), a malware detection system based on permission usage analysis to cope with the rapid increase in the number of Android malware</td>
</tr>
<tr>
<td>5.</td>
<td>Robust Malware Detection for Internet of (Battlefield) Things Devices Using Deep Eigenspace Learning</td>
<td>March 2019</td>
<td>Amin Azmoodeh, Ali Dehghantahna</td>
<td>Firstly they transmute OpCodes into a vector space and apply a deep Eigenspace learning approach to classify malicious and benign applications. They also demonstrate the robustness of our proposed approach in malware detection and its sustainability against junk code insertion attacks</td>
</tr>
<tr>
<td>6.</td>
<td>A Multimodal Deep Learning Method</td>
<td>March 2019</td>
<td></td>
<td>Framework uses various kinds of features to reflect the</td>
</tr>
</tbody>
</table>
for Android Malware Detection Using Various Features

properties of android applications from various aspects, and the features are refined using our existence-based or similarity-based feature extraction method for effective feature representation on malware detection.

IV. RELATED WORK

ANDROID MALWARE SURVEY

The security of android Apps has become an essential issue due to that it becomes active field of research. Like the detection of the primary malicious programs, currently a day’s variety of latest malwares are being detected on a daily basis. This section covers Android malware survey papers. There are many connected work, Author provided one in every of the primary survey on mobile malware and analysed a complete of 46 samples. These samples were of iOS, Symbian and android and that they were collected from 2009 to 2011. Main target of assailant were the golem Smartphone market. There were many reasons to write a mobile malware due to their incentives. Author given this and future incentives and conjointly examined the reason for incentives. Evaluated android malware to search out botnet behaviour. The aim of distinctive botnet behaviour was to search out specific trends and characteristics. These trends had extracted from android code and structure. As a result, Author identified characteristics and explored them in terms of golem botnet invention method. This method includes infection, propagation and execution of malware. Finally this method cause botnet maturity model for golem.

[13]Focused on the Android platform and characterize existing android malware in forty nine distinct malware families. Authors performed a time line study to characterize varied malwares supported their installation method, activation mechanism and nature of payloads. The data set but contains nearly 1260 android malware samples being collected between 2010 and 2011. supported the analysis of 4 antivirus software, they achieved best case detection rate 79.6% and worst case detection rate 20.2%.

[7]Discussed the popularity and analysis of Android malware categories. For analysis authors collected 1485 malware samples from fifty eight malware families. The author designated the characteristic of Android malware from finish users and planned a solution as recommendation to users before installing apps. The suggested result was the ultimate need to mitigate the harm happened with golem users. [3]Suggested that a deep knowledge of characteristics of malware is that the initial step to stop from several unwanted consequences gift within the app. The speedy increase of malicious apps at Google play store created new possibilities of threat for Google and user. Author mentioned the quantity of default security mechanism provided by Google for golem Apps. Author conjointly targeted on properties found in acquainted malware apps, and conjointly reviewed the mitigation technique of android malware.
MALWARE DETECTION TECHNIQUES

This section covers non-android based malware detection literature. Author examined 45 different malware detection techniques and conjointly given the scope to match these detection techniques to determine call rules. These call rules lead to secure app development system. Although quality of these system depends on the utilised technique. The previous malware protection research were not supported malicious activities. They were depended on the practicality limitations of mobile phones. Author mentioned the gap between these 2 and threw some lightweight on gift detection and analysis techniques and their professionals and cons. Author also mentioned however one will improved upon these techniques in current malware detection and analysis techniques [8]. Author showed that answer available for malware detection historically based on signature-based techniques. However these techniques were declined thanks to some obfuscation techniques utilized by malware rese and utilised De-obfuscation and Unpacking technique as anti-evasion approach of malware. De-obfuscation techniques finds the standing of obfuscation and unpacking technique is that the method of analysing at the code that offers the precise data of dynamic behaviour. Author conjointly used “bi-feature technique” rather than static Mono-features analysis [11].

D. MANIFEST features based ANDROID APPS ANALYSIS

This section covers Authors used possible to seek out the operate calls gift in code. The read elf command is employed to extract these operate calls. The obtained call list is correlative with malware executables for classification. Author used Partial call Tree formula, Prims and KNN. Further they showed a combined malware detection approach to boost the results. Authors dissecting the API calls of 940 apps and the causes of over privileging in humanoid apps; author found that app author tried to follow token privilege set however typically failing thanks to errors that could be attributed to too little API documentation. Author of this paper build a weapon to observe over privilege options in humanoid apps.

Author used his weapon “Stowaway” on virtually 940 apps. As a result author found that one third apps were over privileged.

MANIFEST AND CODE BASED DETECTION AND ANALYSIS

Authors introduced deed decompile, which recovers android app source code by utilizing installation image directly. The author of this paper used a static investigated 21 million lines of recovered educational code to style and study of Smartphone apps. It uncovered each dangerous functionality and vulnerabilities like message of phone identifiers, botnet characteristic or use of advertisement libraries by finding out one,100 well-liked free android apps. Droid Mat [12] extracted options were used permissions, preparation of parts, API calls and intent message passing to seek out the behaviour of android apps. Author projected and developed a system referred to as Droid Mat. The extracted the data like requested permission intent message passing are from manifest file and API calls are from code based file. Then applied K-means agglomeration and KNN classification technique. Agglomeration is employed to boost the malware modelling capability and classification is used for classify the application benign and malicious.
V. CONCLUSION

The growing rate of android malware created difficulties in lifetime of android users. User feels insecure like risk like hanging of phone on receiving a decision, personal data stealing (contacts, pictures, video etc.), and huge quantity of bill whereas connecting to web and lots of a lot of. The obtainable android malware detection approaches has not been able to give higher accuracy. Most of approaches are supported permission-set solely that was inadequate to observe new android malware. Few approaches contemplate few code properties however they weren't able to provide sensible accuracy.

VI. REFERENCES


