Behavior-Based Malware Detection System Approach For Mobile Security Using Machine Learning

1st Mrs.Seema Vanjire
Department of Computer Science and Engineering
Sathyabama Institute of Science and Technology
Chennai, India
seemasvanjire@gmail.com

2nd Dr.M.Lakshmi
Department of Computer Science and Engineering
SRM Institute of Science and Technology
Kattangulathur, India
laks@icadsindia.com

2021 International Conference on Artificial Intelligence and Machine Vision (AIMV) | 978-1-6654-4211-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/AIMV53313.2021.9671009

Abstract— **In today's world, mobile security is critical not only for our society but also for each individual. Today, everyone wants their own mobile device, which has resulted in a growth in the number of Android users around the world. Each device with internet access interacts with a variety of applications, resulting in a large number of malware infections or dangers in a mobile home. Our strategy moving forward will be to keep everyone's mobile device secure. So, using machine learning, we've created a model for a behavior-based anomaly detection system from an Android mobile device. We used three machine algorithms in this system to detect malware vulnerabilities based on the behaviour of mobile applications. To determine the accuracy of mobile application behaviour in this system, we employed KNN, Naive Bayes, and a decision tree method. As a result, this technique can be utilised to keep a person's Android mobile secure.**

Keywords— Mobile security, Malware detection, Mobile Computing

I. Introduction

With handheld technology advancing at an exponential rate, almost every advancement in the digital world is receiving more attention than ever before. This can be attributed in large part to the ever-expanding mobile phone ecosystem. There are currently over two billion mobile phone devices in use worldwide, including feature phones and tablets. When it comes to the operating system or software that powers these smartphones, Google's Android has emerged as the clear winner in the ecosystem, with a lion's share of 80 percent, beating out Apple's iOS and Microsoft's Windows Phone. This success and popularity can be attributed to the ease with which Android allows developers to create applications, as well as its open-source nature. With popularity comes a darker side — Android flaws. Because the operating system is an open-source platform, it has become a breeding ground for criminals to develop malware applications that expose security and other flaws, exposing critical information such as user data and privacy.Although Google has made significant efforts to reduce malware applications in recent years, it cannot be said that these negative intentions have been completely mitigated. To detect mobile device behaviour, it is necessary to define the types of actions that can be taken and the various types of data that can be collected, allowing for reliable and valid measurements. The Behavior Approach's concept is to analyse the application statically and the mobile device behaviour at runtime. Some assumptions must be made when designing and implementing a mobile application for this system. The user must have an Android phone; the user must install the application on his phone and then log in to the application. We must also consider some functional requirements about the user and system. To begin application monitoring, the user must first launch the application. The procedure begins by gathering information about all systems and installed applications on an Android device, which is then displayed to the user. The data consists of the number of times the user opens the application, which is referred to as the application opening time. The user spends the most time on that application. Shortcut for launching that application and viewing its details.

II. System Design Flow

1.START
2. SHOW
3.IGNORE
4. HIDE

Fig 1. System Design Flow

The above diagram of the system design flow depicts four functional parts: start, show, ignore, and hide. Each section is explained in detail below.

Start: This is the first step in the design process. Start or open the application in this section to monitor the system as well as all installed application usage.

Show: This is the second stage of the design process. This section displays the application usage, which includes the total time spent by the application and the number of times it has been opened by the user. This section also displays the complete data used by applications such as mobile data and Wi-Fi data usage.

Ignore: This is the third stage of the design process. This section will display a list of applications that have been ignored

Hide: This is the fourth stage of the design process. It will display a list of Hidden applications or applications that are running in the background.

III. Literature Survey

The paper[1] describes a method for protecting mobile devices from malware. The paper[2] proposes a novel malware detection framework for mobile device security. The Support Vector Machine (SVM) machine learning algorithm is demonstrated in paper [3]. Paper[4] provides a clever method for detecting malware in Android applications. The paper[5] highlights malware detection models for Android OS. The paper[6] describes a model for detecting and preventing malicious behaviour in Android applications. The paper[7] proposes a method and modelling framework. The paper[8] proposes a method for modern mobile devices to detect malware with high accuracy. Paper[9] provides a framework for analysing user behaviour based on mobile phone activity.

IV. Malware Detection

There are two types of malware detection in computers: static analysis and dynamic analysis. The same is true for Android OS; static analysis examines an application file's functionality without executing it, whereas dynamic analysis examines the file by running it on a computer or even sandbox tools to investigate the malware's behaviour in depth. Because modern Android malware contains extra elements such as evasion techniques, all recent studies have focused on dynamic analysis in ML. Static analysis typically considers techniques such as packed encryption and mitigates the effects of malware, but it cannot help because it may leave 'traces' vulnerable to attacks.

Botnets are another type of malware that has recently gained popularity. What was once only found in computers has now spread to mobile devices like smartphones. Botnet-affected device networks can act independently, posing a threat to the mobile ecosystem. Botnet attacks are typically carried out without the user's knowledge and are now being deployed on Android-powered smartphones. Regardless of mobile platform, critical data can be stolen via distributed denial of service (DDoS) attacks such as HTTP Flood, Ping Flood, and so on. Although ML detection has emerged to combat DDoS, it has yet to make an impact in the mobile space.

V. Discerning Malware through machine learning

Studies and methods for detecting malware in Android can be traced back to its initial release in 2008. Since then, a plethora of software tools, such as sandboxes and debuggers, have been developed and used for malware analysis. However, with the staggering rise in malware outbreaks in recent years, it is difficult to contain with just these tools. This problem compelled computer scientists and researchers to devise machine learning (ML) methods.

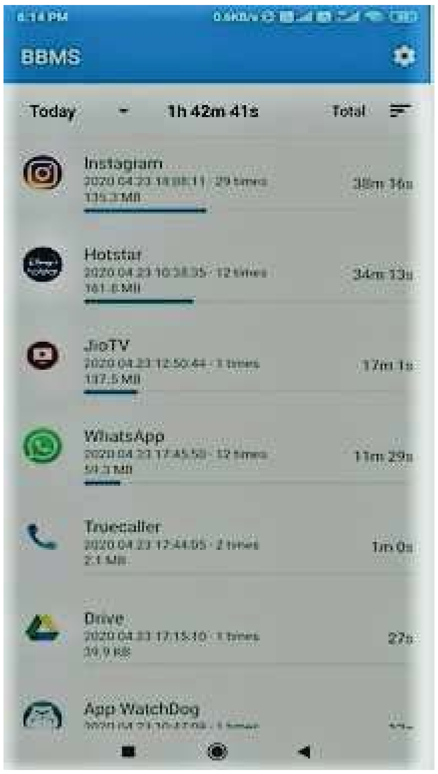
The first studies looked into ML techniques like classification to distinguish between malicious and legitimate applications.Android Package Kits (.apk) files, which are Android application files, were thoroughly tested using ML algorithms to look for malicious software code. These studies also looked for inconsistencies in the code that could leave applications vulnerable to attacks.

The difficulty here is in capturing the precise features for ML in applications. Some studies used support vector machines (SVM) to identify different types of malware classes in order to combat this. In addition, to improve detection, these studies used tools such as the control flow graph (CFG) to represent the application learning flow. These studies demonstrated that feature extraction was made easier than in previous studies. Following research on identifying Android malware using ML, Bayesian classification approaches yielded significant results in terms of detection accuracy.

Computer scientists have also investigated application permissions to see if ML methods have an impact in this area. It was discovered to have fairly good detection accuracies of up to 90%. Some computer developers have even begun to integrate ML in sandboxes, which has the potential to address vulnerabilities in applications' online services as well.

VI. System specification

The Derbin dataset is used to implement this system in Py-charm. This system has some functional requirements. The proposed method can monitor all networks and applications installed on the mobile device. Furthermore, no additional hardware is required. The system's restrictions are stated in this system, such as the fact that if a programme crashes, we will be unable to obtain any relevant information about the installed application. This system has the potential to detect user stress levels in working environments in the future.



6:14 PM 0.6KB/s

BBMS

Today 1h 42m 41s Total

Instagram
2020.04.23 18:08:11 · 29 times
135.3 MB
38m 16s

Hotstar
2020.04.23 10:38:35 · 12 times
161.6 MB
34m 13s

JioTV
2020.04.23 12:50:44 · 1 times
117.5 MB
17m 1s

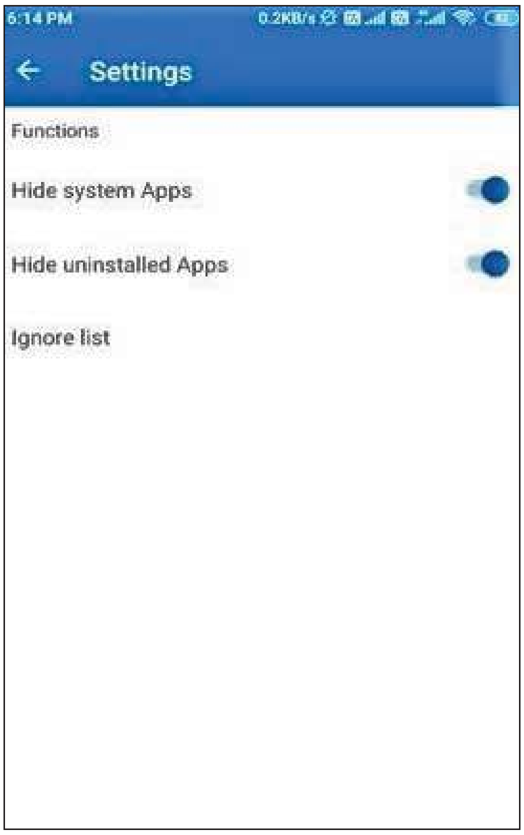
WhatsApp
2020.04.23 17:45:50 · 12 times
59.3 MB
11m 29s

Truecaller
2020.04.23 17:44:05 · 2 times
2.1 MB
1m 0s

Drive
2020.04.23 17:15:10 · 1 times
39.9 KB
27s

App WatchDog

Fig 2. Data collection



6:14 PM 0.2KB/s

Settings

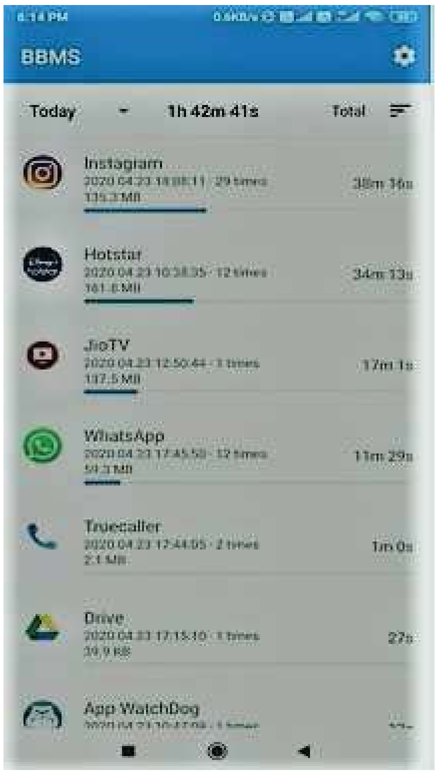
Functions

Hide system Apps

Hide uninstalled Apps

Ignore list

Fig 4. Hide Application



6:14 PM 0.6KB/s

BBMS

Today 1h 42m 41s Total

Instagram
2020.04.23 18:08:11 · 29 times
135.3 MB
38m 16s

Hotstar
2020.04.23 10:38:35 · 12 times
161.6 MB
34m 13s

JioTV
2020.04.23 12:50:44 · 1 times
117.5 MB
17m 1s

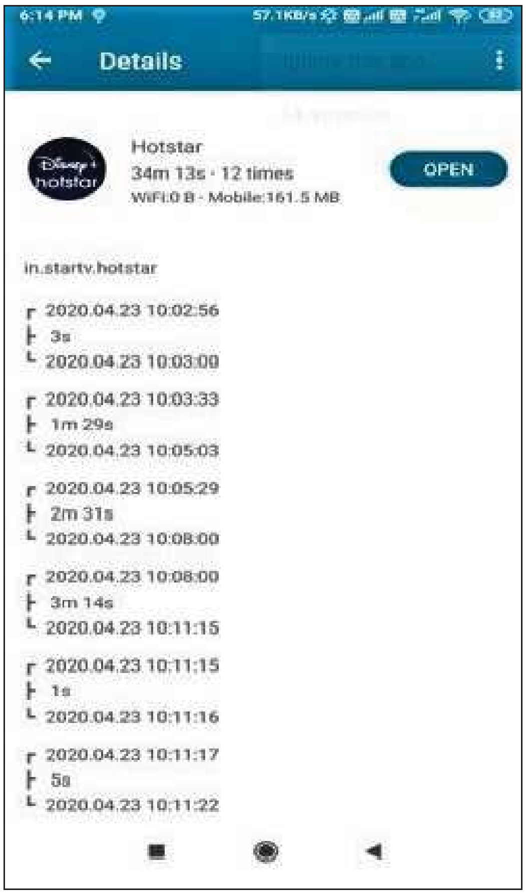
WhatsApp
2020.04.23 17:45:50 · 12 times
59.3 MB
11m 29s

Truecaller
2020.04.23 17:44:05 · 2 times
2.1 MB
1m 0s

Drive
2020.04.23 17:15:10 · 1 times
39.9 KB
27s

App WatchDog

Fig3. Application Permission



6:14 PM 57.1KB/s

Details

Hotstar
34m 13s · 12 times
WiFi:0 B · Mobile:161.5 MB

OPEN

in.startv.hotstar

```
┌ 2020.04.23 10:02:56
├ 3s
└ 2020.04.23 10:03:00

┌ 2020.04.23 10:03:33
├ 1m 29s
└ 2020.04.23 10:05:03

┌ 2020.04.23 10:05:29
├ 2m 31s
└ 2020.04.23 10:08:00

┌ 2020.04.23 10:08:00
├ 3m 14s
└ 2020.04.23 10:11:15

┌ 2020.04.23 10:11:15
├ 1s
└ 2020.04.23 10:11:16

┌ 2020.04.23 10:11:17
├ 5s
└ 2020.04.23 10:11:22
```

Fig 5. Application Usage

Figure 2 in the preceding figures depicts a screenshot of behavior-based malicious application detection for data collection. Figure 3 also depicts a screenshot of malicious application permissions. Similarly, fig.4 depicts an example of how to hide applications. As well as Fig. 5 depicts the use of a malicious application.

VII. Result analysis

Machine learning algorithms such as KNN, Naive Bayes, and Decision Tree are used to implement the system. These algorithms are used to detect malicious behaviour in mobile devices and their accuracy is calculated. The Nave Bayes classifier is the most accurate (97.37 %), followed by KNN (91.99 %) and Decision Tree (89.98 %).The system is implemented using a machine learning algorithm as KNN, Naive Bayes, and Decision Tree machine learning algorithm. Accuracy is calculated by using these algorithms to detect malicious behavior of mobile devices. The Naïve Bayes classifier gives the highest accuracy (97.37%), followed by KNN (91.99%) and Decision Tree (89.98%). Figure 6 depicts the outcome of this conclusion in the form of a graph.

Table1. Result Accuracy

Machine Learning Algorithm	Accuracy
KNN	91.99%
Naïve Bayes	97.37%
Decision Tree	89.98%

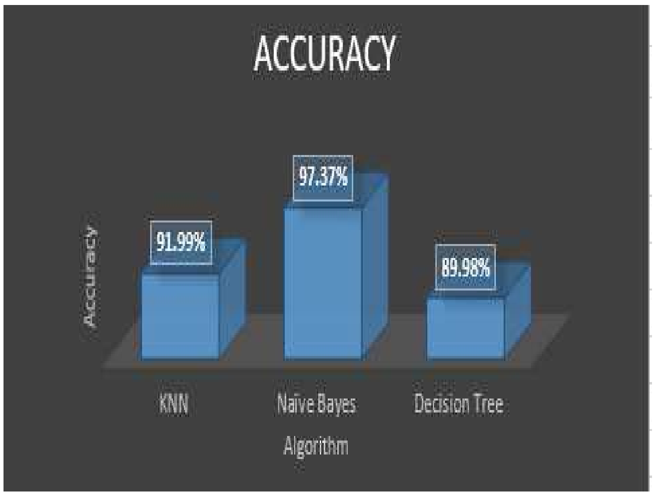


Fig 6. The accuracy achieved through the ML algorithm

VIII. Conclusion

ML models take a proactive approach to malware removal. However, as previously stated, the feature extraction bit requires significant improvement. For example, if features do not match closely in the training, a specific part of malware may be avoided in the output. As a result, a rigid ML framework is always recommended to combat modern shapeshifting malware and the adversarial impact it can have on sensitive data. The following statements round out the approach for this system. In this system, we used KNN, Naive Bayes, and a decision tree algorithm to determine the accuracy of mobile application behaviour. Among these algorithms, the naive Bayes algorithm is the most accurate at detecting malicious behavior from smart Android mobile devices. This system will now be used to keep an individual's Android mobile security up to date. The design of this system will be modified in the future using artificial intelligence and an in-depth learning approach. This type of system dataset may be limited for future work, so we must use a more extensive and diverse dataset.

References

[1] Nayeem Islam, Saumitra Das, Yin Chen, "On-Device Mobile Phone Security Exploits Machine Learning, IEEE Pervasive Computing", vol. 16, Issue 2, pp. 92-96, April-June 2017.

[2] Khurram Majeed, Yangshuo Jing, Dusica Novakovic, Karim Ouazzane, "Behaviour Based Anomaly Detection for Smartphones Using Machine Learning Algorithm", Proceedings of the International Conference on Computer Science and Information Systems (ICSIS2014), pp. 67- 73,2014.

[3] Ashka Shamili, Christian Bauckhage, Tansu Alpcan, "Malware Detection on Mobile Devices Using Distributed Machine Learning", Proceedings of the 20th International Conference on Pattern Recognition, pp. 4348 4351, 2010.

[4] M. Waqar Afridi, Toqeer Ali, Turki Alghamdi, Tamleek Ali, Muhammad Yasar, "Android Application Behavioral Analysis Through Intent Monitoring", Proceedings of the 6th International Symposium on Digital Forensic and Security (ISDFS), pp. 1-8, 2018.

[5] Saba Arshad, Munam Shah, Abdul Wahid, Amjad Mehmood, Housing Song, Hongnian Yu, "SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System", IEEE Access, vol. 6, pp. 4321-4339, January 2018.

[6] A. Saracino, D. Sgandurra, G. Dini, F. Martinelli, "MADAM: E active and efficient behavior-based android malware detection and prevention", IEEE Transactions on Dependable and Secure Computing., vol. 15, no. 1, pp. 83-97, Jan-Feb. 2018.

[7] Zhang H, Yan Z, Yang J, Munguia Tapia E, Crandall.D, "Fingerprint: Privacy-Preserving User Modelling with Multimodal Mobile Device Footprints". Soc Comput Behav-Cult Model Predict Lecture Notes Comput Sci8393:195203.

[8] Singh VK, Freeman L, Lepri B, Pentland. A," Predicting Spending Behaviour using Socio- Mobile Features". In:BioMedCom 2013

[9] Seema Vanjire, Dr. M. Lakshmi, "MDTA: A New Approach of Supervised Machine Learning For Android Malware Detection and Threat Attribution Using Behavioral Reports," Second International Conference On Mobile Computing And Sustainable Informatics, ICMCSI 2021

[10] Seema Vanjire, Dr. M. Lakshmi, "FNN and Auto Encoder Deep Learning-Based Algorithm For Android Cyber Security," International Journal Of Recent Technology And Engineering (IJRTE), Volume 8 Issue 5, Jan 2020.

[11] Seema Vanjire, Sachin Vanjire, "Nested Tunnel Header Compression Protocol In Wireless Network With NET Technology" IEEE Conference ICCSP 2014.