International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278–9960; ISSN(E): 2278–9979
Vol. 9, Issue 5, Jul–Dec 2020; 1–6

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

TWALK SECURITY DETECTION METHOD FOR ANDROID APPLICATION WITH SVM CLASSIFICATION METHOD

Srinivas B K[1] & Sinchana Gowda R[2]

[1]Assistant Professor, Information Science Department, RVCE, Bengaluru, India

[2]Research Scholar, Information Science Department, RVCE, Bengaluru, India

ABSTRACT

This paper deals with the android application security detection method, based on the permissions .Mobile applications create their own security and privacy models through permission-based models. Some applications may request extra permissions that they do not need but may use for suspicious activities. The aim of this study is to identify those spare permissions requested and use this information in the security and privacy approach. To combat serious malware campaigns, we need a scalable malware detection approach that can effectively and efficiently identify malware apps. The proposed method based on metropolis algorithm with twalk is to provide the detection method for the android security of the applications based on the permissions that the applications request from the users during installations. This method helps to speedup detection and also increasing the sampling rate. Accuracy, precisions and recall values are calculated using true and false positive and negativerates depending on which this model is selected to detect the security of the android application.

KEYWORDS: *Attacks, Android Security, Permissions, Methods*

Article History

Received: *10 Jul 2020* **|** ***Revised:*** *14 Jul 2020* **|** ***Accepted:*** *23 Jul 2020*

INTRODUCTION

The widespread use of mobile devices has led to an increase in the ability of users to store critical information such as personal/confidential data and bank account numbers on their mobile devices. Mobile applications create their own security and privacy models through permission-based models. Some applications may request extra permissions that they do not need but may use for suspicious activities. Ill-intentioned people who are aware of this aim to acquire money/reputation illegally make use of this information. In order to prevent unwanted consequences caused by such actions, mobile device manufacturers, mobile application developers, mobile application market owners and governments are engaged in coming up with new precautions such as dangerous permission detection methods. Android OS is susceptible to various security attacks due to its weakness in security.

Why Android Application Security is Important

With the number of android device users increasing year after year, the need for mobile security is also gaining ground. Android phones have now become vulnerable because of the rapid progress in the mobile phone industry and the introduction of cloud services and apps. In other words, android security has yet to reach millions of users, unlike the progress seen in the overall usage of smart phones.

The Two main reasons why android security are important due,

- Android security protects you against malvertisers
- Android security protects your private data

To provide the security, android security detections methods play a important role in safeguarding the android applications.

PRILIMINARIE

In this section, readers are provided with the preliminary details which are necessary to understand the purpose, techniques and key concerns of the various research works. The background of Android permission and Metropolis Algorithm with twalk is introduced.

Permissions

Permissions are an important security mechanism for Android systems. Android permissions are primarily used to limit the use of restricted functionality within an application, as well as component access between applications. Android permissions are divided into normal permissions and dangerous permissions. Normal permissions cover applications that need access to their sandbox, external data or resources. However, there is little risk to user privacy or other application operations. These permissions are granted when the app is installed. The user is no longer asked at runtime. For example: network access, Wi-Fi status, volume settings, etc. Dangerous permissions include areas where the application requires data or resources that involve user privacy information, or that may affect the operation of data stored by users or other applications. For example: Read address book, read and write memory data, get user location, etc. If the application claims that these dangerous permissions are required, the user must be explicitly told atrun time to have the user manually grant them. As shown in Table.

Group	*number*	*Dangerous Permissions*
CONTACTS	3	WRITE_CONTACTS GET_ACCOUNTS READ_CONTACTS
PHONE	7	READ_CALL_LOG READ_PHONE_STATE CALL_PHONE WRITE_CALL_LOG USE_SIP PROCESS_OUTGOING_CALLS ADD_VOICEMAIL
CALENDAR	2	READ_CALENDAR WRITE_CALENDAR
CAMERA	1	CAMERA
SENSORS	1	BODY_SENSORS
LOCATION	2	ACCESS_FINE_LOCATION ACCESS_COARSE_LOCATION
STORAGE	2	READ_EXTERNAL_STORAGE WRITE_EXTERNAL_STORAGE
MICROPHONE	1	RECORD_AUDIO
SMS	5	READ_SMS RECEIVE_WAP_PUSH RECEIVE_SMS RECEIVE_MMS SEND_SMS

Figure 1

Although Google offers dangerous permissions, J. Li et al. [12] study indicates that even normal permissions can be exploited by mal Apps and a few dangerous permissions do not appear in the top-40risky list. W. Wang et al. [1] compared their list of 22significant permissions to the list of 24dangerouspermissionsidentified by Google; they noticed that there are only 8 permissions that appear on both lists. At the same time, the detection technology is extremely difficult.

Statisticized by Statista: As of January 2018, 4.4% of users still use the Android operating system version lower than Android 6.x. This method analyzes the confidence intervals of 24 Android dangerous permissions. Extract dangerous permissions that significantly affect the security of the application.

MCMC

The MCMC method is a Monte Carlo simulation using a Markov chain. The basic principle of the MCMC method is to construct a probability transfer matrix using carefully balanced conditions. Make the target probability the smooth distribution of the probability transfer matrix. There are two methods for constructing the probability transfer matrix: Metropolis-Hasting and Gibbs sampling. In statistics, the Metropolis–Hastings algorithm with twalk is a MCMC method for obtaining a sequence of random samples from aprobability distribution without tuning for which direct sampling is difficult. This method calculates the certainty of Android dangerous permissions through the Metropolis sampling algorithm.[1]

Tuning

"Tuning" broadly describes what happens before sampling. In Bayesian inference, our goal is to calculate expectations over probability distributions. MCMC achieves that by producing samples that are *asymptotically guaranteed* to come from the distribution of interest: the "stationary distribution".

PROPOSEDSECURITYDETECTION METHODS USING METROPOLIS ALGORITHM WITH TWALK

Metropolis Detection Method with Twalk

Metropolist walk is a new general purpose MCMC sampler for arbitrary continuous distributions that requires no tuning.

The t-walk maintains two independent points in the sample space, and all moves are based on proposals that are then accepted with a standard Metropolis-Hastings acceptance probability on the product space. Hence the t-walk is provably convergent under the usual mild requirements. We restrict proposal distributions, or ‘moves’, to those that produce an algorithm that is invariant to scale, and approximately invariant to affine transformations of the state space. Hence scaling of proposals, and effectively also coordinate transformations, that might be used to increase efficiency of the sampler, are not needed since the t-walk’s operation is identical on any scaled version of the target distribution. Four moves are given that result in an effective sampling algorithm.

The t-walk is excellent for initial exploration as it overcomes the need to tune proposals for scale and correlation, which is typically the first difficulty encountered when applying MCMC methods. For a large number of problems the t-walk will allow sufficiently efficient sampling of the target distribution that no recourse to further algorithm developments required. Also, the t-walk will prove use fulin multiple data analyses where details of the posterior distribution depend sufficiently on a particular dataset that adjustment would be required to the proposal in a standard Metropolis-Hastings algorithm, allowing for automatic use of MCMC sampling.

Android applications security detection method based on Metropolis algorithm is designed. This method analyzes Android's 24 dangerous permissions using the Metropolis algorithm, removes uncertainty permissions, and extracts certain permission features. Then, the features are learned and classified through the classification technology of machine learning.

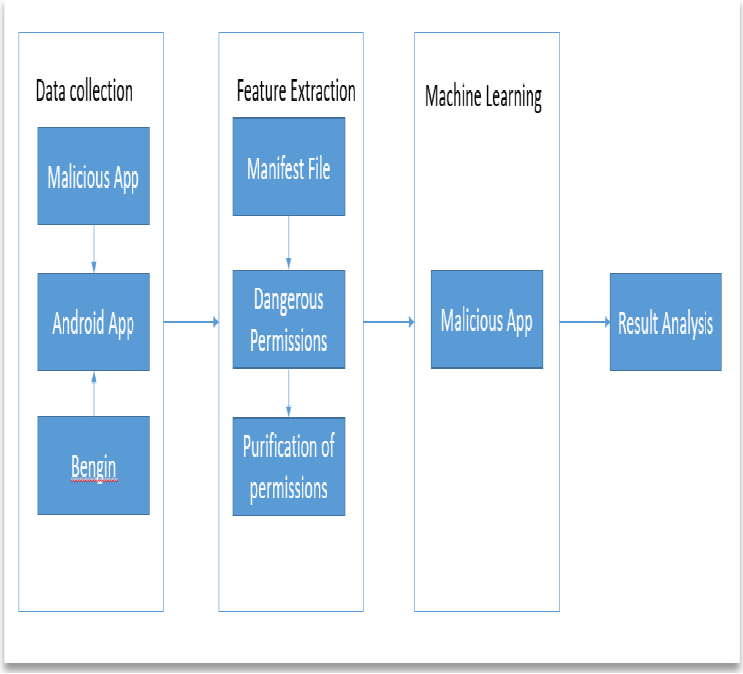

Figure 2: System Model of Detection Method using Metropolis Algorithm with Twalk.

Here we leverage Metropolis algorithm with twalk to get app certain permission. The system model consists of four steps: data set collection, decompiling apk and extract apk's certain permissions, using machine learning to train and test, and result analysis. Figure 2 shows the system model of android security detection method using metropolis twalk algorithm.

Step 1: Collect Apk Samples

Used a python program to crawl benign Apps from the site. Malicious Apps were downloaded in the drebin dataset.

Step 2: Apkdecompile and Feature Extraction

Decompile the apk file with Apk Tool. Extracted the AndroidManifest.xml file containing the permission declaration. Extract the dangerous privilege of 24 Google claims from Android in the AndroidManifest.xml file. Then, using the Metropolis algorithm to extract the certain permissions form alicious app detection.

Step3: Machine Learning-Based Classification

Machine learning is performed by deterministic permissions on the app classification. The data was trained and tested by SVM machine learning classification algorithms. The results indicate that when Support Vector Machine (SVM) is used as the classifier, over 90% of precision, recall, accuracy, and F-measure can be achieved. [2]

Step 4: Experimental Results Evaluation and Analysis

Through machine learning and classification, apps are divided into benign apps and malicious apps. Then, using the Precision, Recall, F-Measure, Accuracy indicators to evaluate the experimental results of machine learning.

RESULTS

For all the above methods mentioned results are evaluated using following assessment methods

Accuracy

Accuracy = (TP+TN)/(TP+FN+FP+TN).That is, the number of samples correctly classified by the detection model divided by all the number of samples.

Precision

Precision=TP/(TP+FP). That is, the proportion of the samples which are classified as malware by the detection model is actually a benign category.

Recall

Recall=TP/(TP+FN). That is, the proportion of the malware samples which are classified as malware by the detection models

CONCLUSIONS

Security checking for Android applications plays a key role in user-safe use of applications. This approach uses the Metropolis algorithm with Twalk to analyze the certainty of24 dangerous permission detections for Android Apps. Here key permissions for malicious application detection are also extracted. For Classifications vmis used to achieve more than90% of precision, recall, accuracy and F measure.

REFERENCES

1. *02 January 2020 ISSN:2576-78282019 IEEE 19th International Conference on Communication Technology (ICCT) "Android Application Security Detection Method Based on Metropolis Algorithm"*
2. *IEEE International Conference on Computer and Communication Engineering Technology (CCET)" A Two-Layered Malware Detection Model Based on Permission for Android"*
3. *Vol14IEEE Transactions on Industrial Informatics, "Significant Permission Identification for Machine-Learning-Based Android Malware Detection".*
4. *2018 IEEE International Conference on Applied System Invention (ICASI), Chiba "Android (Nougats) security issues and solutions"*
5. *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, "Android security issues and solutions,"2017*
6. *Li J, Sun L, Yan Q, et al. Android Malware Detection[J]. IEEE Transactions on Industrial Informatics, 2018, PP(99):1-1.*
7. *Bartel A. Automatically securing permission-based software by reducing the attack surface: an application to Android[C]// International Conference on Automated Software Engineering. IEEE, 2012:274-277.*
8. *D. Geneiatakis, I. N. Fovino, I. Kounelis, P. Stirparo, A permission verification approach for android mobile applications, Computers & Security 49 (2015) 192–205.*

9. *W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: an information flow tracking system for realtime privacy monitoring on smartphones," ACM Transactions on Computer Systems (TOCS), vol.2, no. 2, p. 5, 2014.*
10. *C. Yang, Z. Xu, G. Gu, V. Yegneswaran, and P. Porras, "Droidminer: Automated mining and characterization of fine-grained malicious behaviors in android applications," in European Symposium on Research in Computer Security. Springer, 2014, pp. 16–182*
11. *L. Li, T. F. Bissyand, M. Papadakis, S. Rasthofer, A. Bartel, D. Octeau, J. Klein, T. Le, Static analysis of android Apps: A systematic literature review, Information & Software Technology (2016) pgs. 67–95.*
12. *J. Li, L. Sun, Q. Yan, Z. Li, W. Srisaan, H. Ye, Significant permission identification for machine-learning-based android malware detection, IEEE Transactions on Industrial Informatics 14 (2018)216–225.*