EXPLORING THREATS: FRAMEWORK OF MACHINE LEARNING FOR POSSIBLE REVERSE ENGINEERED ANDROID APP ANALYZATION

[1]Dr M Sravan Kumar Reddy, [2]C Supraja

[1]Associate Professor, [2]M. Tech Student Department of Computer Science

Rajeev Gandhi Memorial College of Engineering and TechnologyNandyal, 518501, Andhra Pradesh, India.

ABSTRACT:

Today, Android is one of the most used operating systems in smartphone technology. This is the main reason, Android has become the favorite target for hackers and attackers. Malicious codes are being embedded in Android applications in such a sophisticated manner that detecting and identifying an application as a malware has become the toughest job for security providers. In terms of ingenuity and cognition, Android malware has progressed to the point where they're more impervious to conventional detection techniques. Approaches based on machine learning haveemerged as a much more effective way to tackle the intricacy and originality of developing Android threats. They function by first identifying current patterns of malware activity and then using this information to distinguish between identified threats and unidentified threats with unknown behavior. This research paperuses Reverse Engineered Android applications' features and Machine Learningalgorithms to find vulnerabilities present in Smartphone applications. Our contribution is twofold. Firstly, we propose a model that incorporates more innovative static feature sets with the largest current datasets of malware samples than conventional.methods. Secondly, we have used ensemble learning with machine learning algorithms such as AdaBoost, SVM, etc. to improve our model's performance. Our experimental results and findings exhibit 96.24% accuracy to detect extracted malware from Android applications, with a 0.3 False Positive Rate (FPR). The proposed model incorporates ignored detrimental features such as permissions, intents, API calls, and so on, trained by feeding a solitary arbitrary feature, extracted by reverse engineering as an input to the machine.

1. INTRODUCTION

To this degree, it is guaranteedthat mobile devices are an integral part of most people's daily lives. Furthermore, Android now controls the vast majority of mobile devices, with Android devices accounting for an average of 80% of theglobal market share over the past years. With the ongoing plan of Android to a growing range of smart phones andconsumers around the world, malware targeting Android devices has increased as well. Since it is an open-source operating system, the level of danger it poses, with malware authors and programmers implementing unwanted permissions, features and application components in Android apps. The option to expand its capabilities with third-party software is also appealing, but this capability comes with the risk of malicious device attacks. When the number of smart phone apps increases, so does the security problem

with unnecessary access to different personal resources. As a result, the applications are becoming more insecure, and they are stealing personal information, SMS frauds, ransom ware, etc.

In contrast to static analysis methods such as a manual assessment of AndroidManifest.xml, source files and Dalvik Byte Code and the complex analysis of a managed environment to study the way it treats a program, Machine Learning includes learning the fundamental rules and habits of the positive and malicious settings of apps and then data-venabling. The static attributes derived from an application areextensively used in machine learning methodologies and the tedious task of this can be relieved if the static features of reverse-engineered Android Applications are extracted and use machine learning SVM algorithm, logistic progression, ensemble learning and other algorithms to help train the model for prediction of these malware applications [1].

Machine learning employs a range of methodologies for data classification. SVM (Support Vector Machine) is a strong learner that plots each data item as a point in n-dimensional space (where n denotes the number of features you have), with the value of each feature becoming the vector value. Then it executes classification by locating the hyper-plane that best distinguishes the two groups, leading to an improvement identification property for any two parameters. Conversely, boosting or ensemble techniques like Adaboost are assigned higher weights to rectify the behavior of misclassified variables in conjunction with other machine algorithms. When combined alongside weak classifiers, our preliminary model benefits from deploying such models since they have ahigh degree of precision or classification.[2], [3], [4], supports classifiers in their system models to find the highest accuracy. Although using ensemble or strong classifiers can cause problems like multi collinearity, which in a regression model,occurs when two or more independentvariables are strongly associated with one another. In multivariate regression, this indicates that one regression analysis may be forecasted from another independent variable. This scope of the study can be presented as a detection journal analysis itself and can present several experimentations and results based on machine learning models [5], [6].

When an app has access to a resource in the most recent versions of Android OS, it must ask the OS for approval, and the OS will ask the user if they wish to grant or refuse the request via a pop-up menu. Many reports have been performed on the success of this resource management approach. The studies showed consumers made decisions by giving all requested access to the applications to their privileges requests [7]. In contrast to this, over 70% of Android mobile applicationsseek extra access that is not needed. They also sought a permit that is not needed forthe app to run. A chess game that asks for
photographs or requests for SMS and phone call permits, or loads unwanted packages arean example of an extra requested authorization. So, trying to assess an app's vindictiveness and not understanding the app is a tough challenge. As a result, successful malicious app monitoring will provide extra information to customers toassist them and defend them from information disclosure [8]. Figure 1 elaborates the android risk framework through the Google Play platform, which is then manually configured by the android device developers.

Contrary to other smart phone formats, such as IOS, Android requires users to access apps from untrusted outlets likefile- sharing sites or third-party app stores. The malware virus problem has become so severe that 97 % of all Smartphone malware now targets Android phones. In a year, approximately 3.25 million new malware Android applications are discovered as the growth of smartphones increases. This loosely amounts to a new malware android version being introduced every few seconds [9]. The primary aim of mobile malware is to gain entrance to user data saved on the computer and user information used inconfidential financial activities, such as banking. Infected file extensions, files received via Bluetooth, links to infected code in SMS, and MMS application links are all ways that mobile malware canpropagate [10]. There are some strategies forlocating apps that need additional features. Hopefully, by using these techniques, it would be possible to determine whether the applications that were flagged as questionable and needed additionalauthorization are malicious.

Static analysis methodologies are the most fundamental of all approaches. Until operating programs, the permissions and source codes are examined [11]. For many machine learning tasks, such as enhancing predictive performance or simplifying complicated learning problems, ensemble learning is regarded as the most advanced method. It enhances a singlemodel's prediction performance by training several models and combining their predictions. Boosting, bagging, and random forest are examples of common ensemblelearning techniques [12]. In summary, the main contributions of our study are as follows:

1) We present a novel subset of features for static detection of Android malware, which consists of seven additional selected feature sets that are using around 56000 features from these categories. On a collection of more than 500k benign and malicious Android applications and the highest malware sample set than any state-of-the-art approach, we assess their stability.The results obtain a detection increase inaccuracy to 96.24 % with 0.3% false- positives.

2) With the additional features, we have trained six classifier models or machine learning algorithms and also implemented a Boosting ensemble learning approach (AdaBoost) with a Decision Tree based on the binary classification to enhanceour prediction rate.

3) Our model is trained on the latest and large time aware samples of malware collected within recent years including the latest Android API level than state-of-the-artapproaches. This research paper incorporates binary vector mapping for classification by allocating 0 to maliciousapplications and 1 for non-harmful and for predictive analysis of each application fed tothe model implemented in the study. Thetechnique eases the process by reducing fault predictive errors. Figure 2 shows the procedure for a better understanding of the concept applied later in our study. The paperpasses both the categories of applicationsthrough static analysis and then is furtherprocessed for feature extraction. We presented features in 0’s and 1’s afterextraction. Matrix displays the extraction characteristics of each application used inthe dataset. There are major issues to be addressed to incorporate our strategy. High measurements of the features will make it difficult to identify malware in many real- world Android applications. Certain featuresoverlap with innocuous apps and malware [13]. In comparison,

the vast number of features will cause high throughput computing. Therefore, we can learn from thefeatures directly derived from Android apps,the most popular and significant features.The paper implements prediction models and various computer ensemble teaching strategies to boost and enhance accuracy to resolve this problem [14]. Feature selection is an essential step in all machine-based learning approaches. The optimum collection of features will not only help boost the outcomes of tests but will also help to reduce the compass of most achine-based learning algorithms [15].

2. LITERATURE SURVEY

''Android malware detection throughmachine learning techniques: A review,'' The open source nature of AndroidOperating System has attracted wider adoption of the system by multiple types of developers. This phenomenon has furtherfostered an exponential proliferation of devices running the Android OS into different sectors of the economy. Although this development has brought about great technological advancements and ease ofdoing businesses (e-commerce) and socialinteractions, they have however become strong mediums for the uncontrolled rising cyber attacks and espionage against businessinfrastructures and the individual users ofthese mobile devices. Different cyber attacks techniques exist but attacks through malicious applications have taken the leadaside other attack methods like social engineering. Android malware have evolvedin sophistications and intelligence that theyhave become highly resistant to existingdetection systems especially those that aresignature-based. Machine learningtechniques have risen to become a morecompetent choice for combating the kind ofsophistications and novelty deployed byemerging Android malwares. The modelscreated via machine learning methods workby first learning the existing patterns ofmalware behavior and then use thisknowledge to separate or identify any suchsimilar behaviour from unknown attacks.This paper provided a comprehensive review of machine learning techniques andtheir applications in Android malware detection as found in contemporary literature.</p

''Geometric feature-based facialexpression recognition in image sequencesusing multi-class AdaBoost and supportvector machines,"

Facial expressions are widely used in the behavioral interpretation of emotions, cognitive science, and social interactions. In this paper, we present a novel method for fully automatic facial expression recognition in facial image sequences. As the facial expression evolves over time facial landmarks are automatically tracked in consecutive video frames, using displacements based on elastic bunch graph matching displacement estimation. Feature vectors from individual landmarks, as well as pairs of landmarks tracking results are extracted, and normalized, with respect tothe first frame in the sequence. The prototypical expression sequence for eachclass of facial expression is formed, by taking the median of the landmark tracking results from the training facial expression sequences. Multi-class AdaBoost with dynamic time warping similarity distance between the feature vector of input facialexpression and prototypical facial expression, is used as a weak classifier to select the subset of discriminative feature vectors. Finally, two methods for facial expression recognition are presented, either by using multi-class AdaBoost with dynamic time warping, or by using support vector machine on the boosted feature vectors. The results on the Cohn-Kanade (CK+) facial expression database show a recognition accuracy of

95.17% and 97.35% using multi-class AdaBoost and support vector machines, respectively.

''AdaBoost for feature selection, classification and its relation with SVM, areview,''

In order to clarify the role of AdaBoost algorithm for feature selection, classifier learning and its relation with SVM, this paper provided a brief introduction to theAdaBoost which is used for producing a strong classifier out of weak learners firstly. The original adaptive boosting algorithm and its application in face detection and facial expression recognition are reviewed. In pattern classification domain, support vector machine has been widely used and shows promising performance. However, it is expensive in terms of time-consuming. A sort of cascaded support vector machines architecture is capable of improving the classification accuracy based on AdaBoost boosting algorithm, namely, AdaboostSVM. It applied boosting algorithm to feature selection and classifier learning for support vector machine classification and it has achieved approved performance through some researcher's pioneering work.

''Dynamic financial distress predictionwith concept drift based on time weighting combined with Adaboostsupport vector machine ensemble,' Dynamic financial distress prediction(DFDP) is important for improvingcorporate financial risk management. However, earlier studies ignore the time weight of samples when constructing ensemble FDP models. This study proposes two new DFDP approaches based on time weighting and Adaboost support vector machine (SVM) ensemble. One is the double expert voting ensemble based on Adaboost-SVM and Timeboost-SVM (DEVE-AT), which externally combines the outputs of an error-based decision expert and a time-based decision expert. The other is Adaboost SVM internally integrated with time weighting (ADASVM-TW), which uses a novel error-time-based sample weight updating function in the Adaboost iteration. These two approaches consider time weighting of samples in constructing Adaboost-based SVM ensemble, and they are more suitable for DFDP in case of financial distress concept drift. Empirical experiment is carried out with sample dataof 932 Chinese listed companies' 7 financialratios, and time moving process is simulatedby dividing the sample data into 13 batches with one year as time step. Experimentalresults show that both DEVE-AT and ADASVM-TW have significantly better DFDP performance than single SVM, batch- based ensemble with local weighted scheme, Adaboost-SVM and Timeboost-SVM, and they are more suitable for disposing concept drift of financial distress.

3. EXISTINGSYSTEM

The methods proposed in this related work contribute to key aspects and a higher predictive rate for malware detection. Certain research has focused on increasing accuracy, while others have focused on providing a larger dataset, some have been implemented by employing various feature sets, and many studies have combined all of these to improve detection rate efficiency. In

[21] the authors offer a system for detecting Android malware apps to aid in the organization of the Android Market. The proposed framework aims to provide amachine learning-based malware detectionsystem for Android to detect malware apps and improve phone users' safety and privacy. This system monitors different

permission-based characteristics and events acquired from Android apps and examines these features employing machine learning classifiers to determine if the program is good ware or malicious.

The paper uses two datasets with collectively 700 malware samples and 160 features. Both datasets achieved approximately 91% accuracy with Random Forest (RF) Algorithm. [22] Examines 5,560 malware samples, detecting 94 % of the malware with minimal false alarms, where the reasons supplied for each detection disclose key features of the identified malware. Another technique [23] exceeds both static and dynamic methods that rely on system calls in terms of resilience. Researchers demonstrated the consistency of the model in attaining maximum classification performance and better accuracy compared to two state-of-the-art peer methods that represent both static and dynamic methodologies over for nine years through three interrelated assessments with satisfactory malware samples from different sources. Model continuously achieved 97% F1- measure accuracy for identifyingapplications or categorizing malware.

[24] The authors present a unique Android malware detection approach dubbed Permission- based Malware Detection Systems (PMDS) based on a study of 2950 samples of benign and malicious Androidapplications. In PMDS, requested permissions are viewed as behavioral markers, and a machine learning model is built on those indicators to detect new potentially dangerous behavior in unknown apps depending on the mix of rights they require. PMDS identifies more than 92–94%of all heretofore unknown malware, with a false positive rate of 1.52–3.93%.

The authors of this article [25] solely use the machine learning ensemble learning method Random Forest supervised classifier on Android feature malware samples with 42 features respectively. Their objective was to assess Random Forest's accuracy in identifying Android application activity as harmful or benign. Dataset 1 is built on 1330 malicious apk samples and 407 benign ones seen by the author. This is based on the collection of feature vectors for each application. Based on an ensemble learning approach, Congyi proposes a concept in [26] for recognizing and distinguishing Android malware.

Disadvantages

- The system is not implemented MACHINELEARNINGALGORITHM AND ENSEMBLELEARNING.
- The system is not implemented Reverse Engineered Applicationscharacteristics.

4. PROPOSED SYSTEM

1) We present a novel subset of features for static detection of Android malware, which consists of seven additional selected feature sets that are using around 56000 features
from these categories. On a collection of more than 500k benign and malicious Android applications and the highest malware sample set than any state-of-the-art approach, we assess their stability. The results obtain a detection increase in accuracy to 96.24 % with 0.3% false- positives.

2) With the additional features, we havetrained six classifier models or machine learning algorithms and also

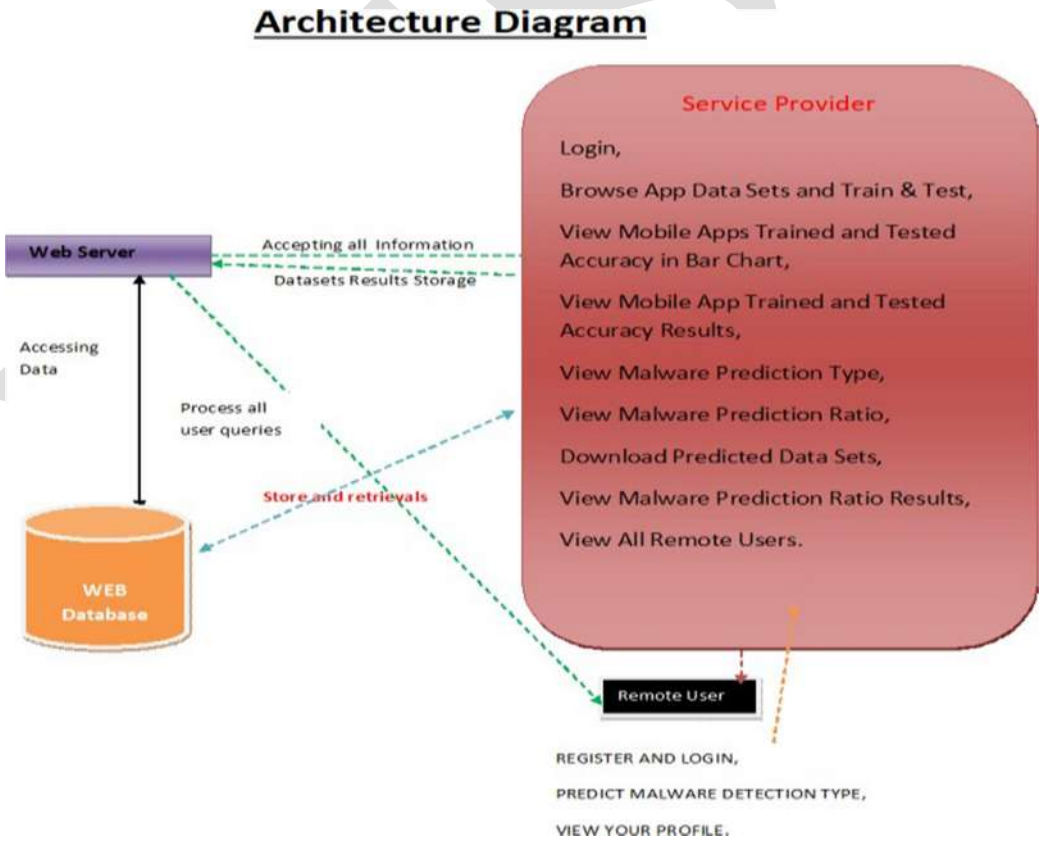
implemented a Boosting ensemble learning approach (AdaBoost) with a Decision Tree based on the binary classification to enhance our prediction rate. 3) Our model is trained on the latest and large time aware samples of malware collected within recent years including the latest Android API level than state-ofthe-art approaches.

Advantages

- The proposed system chooses the characteristics based on their capability to display all data sets. Enhanced efficiency by reducing the dataset size and the hours wasted on the classification process introduces an effective function selection process.
- The system used in this study also incorporates larger feature sets for classification. Although this problemarises in machine learning quite often to some extent choosing the type of model for detection or classification can highly impact the high dimensionality of the data beingused.

User name, email, address and adminauthorizes the users.

5. SYSTEM ARCHITECTURE



Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, PREDICT MALWAREDETECTION TYPE, VIEW YOUR PROFILE.

6. IMPLEMENTATIONModules

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Login, Browse AppData Sets and Train & Test, View Mobile Apps Trained and Tested Accuracy in Bar Chart, View Mobile App Trained and Tested Accuracy Results, View Malware PredictionType, View Malware Prediction Ratio, Download Predicted Data Sets, ViewMalware Prediction Ratio Results, View All Remote Users.

View and Authorize Users

In this module, the admin can view the listof users who all registered. In this, the admin can view the user's details such as,

7. SNAP SHOTS

8. CONCLUSION

In this research, we devised a frameworkthat can detect malicious Android applications. The proposed technique takes into account various elements of machinelearning and achieves a 96.24% in identifying malicious Android applications. We first define and pick functions to captureand analyze Android apps' behavior, leveraging reverse application engineeringand AndroGuard to extract features into binary vectors and then use python buildmodules and split shuffle functions to train the model with benign and malicious datasets. Our experimental findings show that our suggested model has a false positive rate of 0.3 with 96% accuracy in the given environment with an enhanced and largerfeature and sample sets. The study also discovered that when dealing with classifications and high-dimensional data, ensemble and strong learner algorithms perform comparatively better. The suggested approach is restricted in terms of static analysis, lacks sustainability concerns, and fails to address a key multi collinearity barrier. In the future, we'll consider model resilience in terms of enhanced and dynamic features. The issue of dependent variables orhigh inter correlation between machine algorithms before employing them is also a promising field.

REFERENCES

[1] A. O. Christiana, B. A. Gyunka, and A. Noah, ―Android Malware Detection through Machine Learning Techniques: A Review,‖ Int. J. Online Biomed. Eng. IJOE, vol. 16, no. 02, p. 14, Feb. 2020, doi: 10.3991/ijoe.v16i02.11549.

[2] D. Ghimire and J. Lee, ―Geometric Feature-Based Facial Expression Recognition in Image

Sequences UsingMulti-Class AdaBoost and Support Vector Machines,‖ Sensors, vol. 13, no. 6, pp. 7714–7734, Jun. 2013, doi: 10.3390/s130607714.

[3] R. Wang, ―AdaBoost for Feature Selection, Classification and Its Relation with SVM, A Review,‖ Phys. Procedia, vol. 25, pp. 800–807, 2012, doi: 10.1016/j.phpro.2012.03.160.

[4] J. Sun, H. Fujita, P. Chen, and H. Li,
―Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble,‖ Knowl.-Based Syst., vol. 120, pp. 4–14, Mar. 2017, doi: 10.1016/j.knosys.2016.12.019.

[5] A. Garg and K. Tai, ―Comparison of statistical and machine learning methods in modelling of data with multicollinearity,‖ Int. J. Model. Identif. Control, vol. 18, no. 4,p. 295, 2013, doi: 10.1504/IJMIC.2013.053535.

[6] C. P. Obite, N. P. Olewuezi, G. U. Ugwuanyim, and D. C. Bartholomew,
―Multicollinearity Effect in Regression Analysis: A Feed Forward Artificial Neural Network Approach,‖ Asian J. Probab. Stat., pp. 22–33, Jan. 2020, doi: 10.9734/ajpas/2020/v6i130151.

[7] W. Wang et al., ―Constructing Features for Detecting Android Malicious Applications: Issues, Taxonomy and Directions,‖ IEEE Access, vol. 7, pp. 67602–67631, 2019, doi:10.1109/ACCESS.2019.2918139.

[8] B. Rashidi, C. Fung, and E. Bertino,
―Android malicious application detection using support vector machine and active learning,‖ in 2017 13th International Conference on Network and Service Management (CNSM), Tokyo, Nov. 2017, pp. 1–9. doi: 10.23919/CNSM.2017.8256035.

[9] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an,and H. Ye, ―Significant Permission Identification for Machine-Learning-Based Android Malware Detection,‖ IEEE Trans. Ind. Inform., vol. 14, no. 7, pp. 3216–3225, Jul. 2018, doi: 10.1109/TII.2017.2789219.

[10] G. Suarez-Tangil, J. E. Tapiador, P.Peris-Lopez, and J. Blasco, ―Dendroid: A text mining approach to analyzing and classifying code structures in Android malware families,‖ Expert Syst. Appl., vol. 41, no. 4, pp. 1104–1117, Mar. 2014, doi: 10.1016/j.eswa.2013.07.106.

[11] M. Magdum, ―Permission based Mobile Malware Detection System using Machine Learning Techniques,‖ vol. 14, no. 6, pp. 6170–6174, 2015.M. Qiao, A. H. Sung, and Q. Liu,
―Merging Permission and API Features for Android Malware Detection,‖ in 2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI),Kumamoto, Japan, Jul. 2016, pp. 566–571. doi: 10.1109/IIAI- AAI.2016.237.

[12] D. O. Sahin, O. E. Kural, S. Akleylek, and E. Kilic, ―New results on permission based static analysis for Android malware,‖ in 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, Mar. 2018, pp. 1–4. doi: 10.1109/ISDFS.2018.8355377.

[13] A. Mahindru and A. L. Sangal,

–MLDroid—framework for Android malware detection using machine learning techniques,‖ Neural Comput. Appl., vol. 33, no. 10, pp. 5183–5240, May 2021, doi: 10.1007/s00521-020-05309-4.

[14] X. Su, D. Zhang, W. Li, and K. Zhao,

–A Deep Learning Approach to Android Malware Feature Learning and Detection,‖ in 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, China, Aug. 2016, pp. 244–251. doi: 10.1109/TrustCom.2016.0070.