Volume 6, Issue 3, March 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Generating Malware Summary Risk for Android Mobile Applications

S. Santhi
Computer Science & Periyar University,
Tamilnadu, India

***Abstract:** This paper investigates permission-based risk signals that use the rarity of critical permissions and pairs of critical permissions. In this approach, initially reported in permissions that have significant security or privacy impact are identified as critical, and if an app requests a critical permission that is rarely requested by apps in the same category as the app, the app is labelled as risky. Using a binary risk signal, i.e., labelling each app as either risky or not risky, however have significant limitations. A binary risk signal forces the designer to draw a line somewhere, when no line may be correct. This proposes the concept of risk scoring functions. Given a risk scoring function, one can construct a risk signal by choosing a threshold above which the signal is raised. However, believe that it is better to use a risk scoring function for risk communication in the following way. Given this function, one can compute a risk ranking for each app, identifying the percentile of the app in terms of its risk score.*

***Keywords:** Risk Signal, Malware, Permission, Bayesian Risk Score, Simple Risk Score*

I. INTRODUCTION

1.1 Objectives

To provide solution that leverages a method to assign a risk score to each app before installing it and display a summary of that information to users in friendly manner which is easy to understand. So that user can identify potentially risky apps. To provide solution that leverages a method to assign a risk score to each installed app and display a summary of that information to users in friendly manner which is easy to understand. So that user can identify potentially risky apps. To provide the solution for Pileup flaws, by sending notification to users whenever the apps gets auto update and gains some extra permission without user consent. This will alert the user about apps abnormal behavior and he can accordingly decide whether to uninstall or keep the app. To provide the functionality to uninstall the selected app if user finds it malicious.

1.2 Why We Use Mobile Computing?

- Checking email.
- Sending SMS, MMS via mobile phones.
- Continuous access to wireless network service.
- Flexible communication btw people.
- Real time business to employee communication.
- Handheld computers without cables.

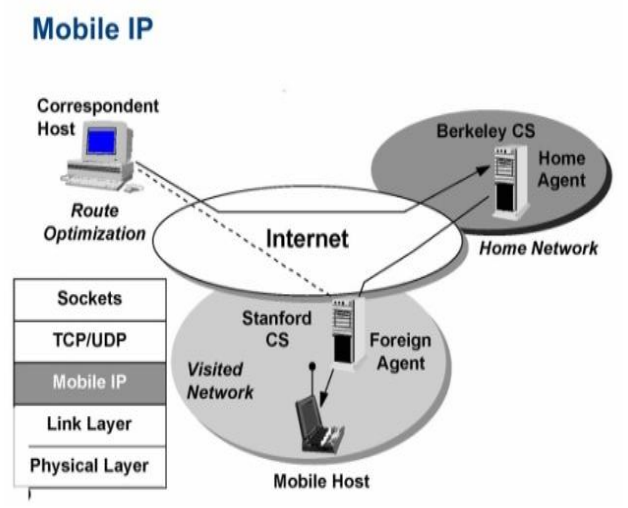


Fig: 1 Transformation of Mobile IP

- Mobile IP is an Internet Engineering Task Force (IETF) standard communications protocol that is designed to allow mobile device users to move from one network to another while maintaining their permanent IP address.
- Request for Comments (RFC), Mobile IP is an enhancement of the Internet Protocol (IP) that adds mechanisms for forwarding Internet traffic to mobile devices when they are connecting through other than their home network.
- Mobile computing is a form of human computer interaction by which a computer is expected to be transported during normal usage.
- Mobile computing has three aspects:
  (i) Mobile Communication
  (ii) Mobile Hardware
  (iii) Mobile Software

1.3 Device Components

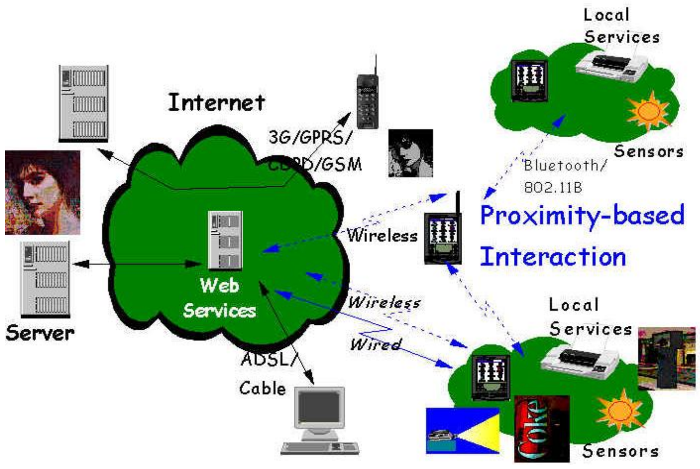


Fig 2: Proximity Based Interaction

- Mobile computing refers to using the small portable and hand held computing devices such as PDA, laptops, mobile phones, MP3 players, digital cameras, tablet PC and Palmtops in a wireless enabled network.
- In recent years the growth of mobile broadband and mobile broadband pay as you go devices have meant that many more people and business can take advantage of mobile computing.

II. LITERATURE SURVEY

2.1 Privacy and Usability

Information privacy is issues for users of all types of electronic devices. With regard to smart phones, users are more concerned with privacy on their phones than on computers, and they especially worry about the threat of malicious apps . Among the recommendations made by Chin et al. [37] was to provide “new security indicators in Smartphone application markets to increase user trust in their selection of applications”. The addition of new security indicators not only may decrease the frequency of risky user behaviors, but it may also facilitate the use of smart phones for online transactions by more individuals. The popularity and adoption of Smartphone’s has greatly stimulated the spread of mobile malware, especially on the popular platforms such as Android. In light of their rapid growth, there is a pressing need to develop effective solutions. However, our defense capability is largely constrained by the limited understanding of these emerging mobile malware and the lack of timely access to related sample. We focus on the Android platform and aim to systematize or characterize existing Android malware.Despite this need, studies of various security and privacy measures have shown their usability is typically deficient [38],which often leads to user resistance .Usability of security mechanisms has been studied in contexts other than mobile platforms. Biddle et al. [39] laid out some general ground rules concerning the content of security dialogs; e.g., avoid unfamiliar terms, lengthy messages and misleading or confusing wordings.

2.2 Mobile OS Updating

Mobile operating systems are evolving quickly. Every a few months, major updates or new overhauls of entire systems are made available, bringing to mobile users brand new apps and enriched functionalities.Recent years have witnessed incredible growth in the popularity and prevalence of smart phones. A flourishing mobile application market has evolved to provide users with additional functionality such as interacting with social networks, games, and more. Mobile applications may have a direct purchasing cost or be free but ad-supported. Unlike in-browser ads, the privacy implications of ads in Android applications has not been thoroughly explored. We start by comparing the similarities and differences of in-browser ads and in-app ads. The reliability of patch installation process has never been called into question. For a mobile system, this update process tends to be more complex, due to its unique security model that confines individual apps within their sandboxes and the presence of a large amount sensitive user data (e.g., contacts, social relations, financial information, etc.) within those apps’ sandboxes.

2.3 Risk Identifying

Users make many decisions that affect the overall state of security of any system with which they interact. For security and privacy, most of these decisions relate to the risk to which the individual or system is exposed. Consequently, improving security decisions by users involves taking into consideration factors that influence a user's risk perception and decision making [38]. As mobile devices become increasingly popular for personal and business use it is becoming increasingly more important to provide users with the ability to understand and control the benefit and risk of running apps on these devices. Mobile devices contain both traditional types of private data, such as contacts, email, and credit card numbers, and new types of resources, including accurate relocation, audio recording, and making phone calls or sending premium SMS messages, all while maintaining constant internet connectivity on high speed wireless networks.

2.4 Security of Android

We show how an attacker can launch malware onto a large number of smart phone users by plagiarizing Android applications and by using elements of social engineering to increase infection rate. Our analysis of a dataset of 158,000 smart phone applications meta information indicates that 29.4% of the applications are more likely to be plagiarized. We propose three detection schemes that rely on syntactic fingerprinting to detect plagiarized applications under different levels of obfuscation used by the attacker. When browsing a specific app from the Google Play website, a user is able to see details about the app via a series of tabs at the top of the page. In addition to an overview, user reviews, and 'what's new' section, one of these tabs presents the permission information. When an app has been selected for installation, permissions are displayed before the user confirms installation. When app installation is performed directly on the device, there is a Play Store app which allows users to find and install new apps. The options and information are the same as on the website, with the primary difference being that the screen may be smaller and so when information is displayed, including permissions, a user has to make more of an effort to view that information.

2.5 Risk Communication

An effective risk communication approach for Android could provide an incentive for developers to reduce the number of permissions requested by apps, similar to UAC's impact with Windows software developers. By highlighting requested permissions of apps, such risk communication could potentially change user behavior and drive consumption.to apps with fewer permissions, thereby creating a feedback loop to developers and having a positive effect on the app ecosystem. Signature-based malware detection tools such as Lookout Security, Norton Mobile Security and Bit Defender Mobile Security detect applications that contain malware. However, they do not detect plagiarized applications that use legitimate permissions, users will still be infected before an attack signature is learned, and the number of infected users can be large due to the centralized nature of markets. Information leakage detection techniques based on taint analysis, access control policies, and kernel modifications protect against stealing critical user information.

III. METHODOLOGY

3.1 Naive Bayes Model

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all Naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 3" in diameter.

3.2 Bayesian Risk Score

In the previous section we looked at risk signals that operate as a binary signal for some risk that might be present from an app. While this appears to work reasonably well, the binary signal approach is inherently limited. A risk score can beused to compare several different applications and understand a relative risk of installing one app versus another. In this section we describe desirable characteristics of a risk score and then present details for several techniques to generate a risk score for Android applications based on the permissions they request. While the focus of this work is on permissions, the idea of generating a risk score can be extended to account for other features such as source code, developer information, user reviews, privacy policies and various other attributes related to an app.

3.3 Simple Risk Scoring Method

We now show that risk ranking functions based on BNB and PNB can be viewed as using permission rarity to rank apps, and is thus closely related to risk signals discussed in Section 4. When risk scores are used to rank apps, using the probabilities (where lower probability means higher risk) is equivalent to using the negative logarithm of the probabilities (where larger value means higher risk). A Naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness and diameter features. For some types of probability models, Naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for Naive Bayes models uses the method of maximum likelihood; in other words, one can work with the Naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite their naive design and apparently oversimplified assumptions, Naive Bayes classifiers have worked quite well in many complex real-world situations .An analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of Naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests. An advantage of Naive Bayes is that it only requires a small amount of training data to estimate the parameters necessary for classification.

3.4 Naive Bayes Classification Algorithm

The Naive Bayes algorithm is a classification algorithm based on Bayes' theorems, and provided by SQL Server Analysis Services for use in predictive modelling. The word naive in the name Naive Bayes derives from the fact that the algorithm uses Bayesian techniques but does not take into account dependencies that may exist. This algorithm is less computationally intense than other algorithms, and therefore is useful for quickly generating mining models to discover relationships between input columns and predictable columns. We can use this algorithm to do initial exploration of data, and then later you can apply the results to create additional mining models with other algorithms that are more computationally intense and more accurate.

How the Algorithm Works ?

The Naive Bayes algorithm calculates the probability of every state of each input column, given each possible state of the predictable column. To understand how this works, use the Naive Bayes Viewer in SQL Server Data Tools (SSDT) (as shown in the following graphic) to visually explore how the algorithm distributes states. Here, the Naive Bayes Viewer lists each input column in the dataset, and shows how the states of each column are distributed, given each state of the predictable column. We would use this view of the model to identify the input columns that are important for differentiating between states of the predictable column. For example, in the row for Commute Distance shown here, the distribution of input values is visibly different for buyers vs. non-buyers. What this tells you is that the input, Commute Distance = 0-1 miles, is a potential predictor. The viewer also provides values for the distributions, so We can see that for customers who commute from one to two miles to work, the probability of them buying a bike is 0.387, and the probability that they will not buy a bike is 0.287. In this example, the algorithm uses the numeric information, derived from customer characteristics (such as commute distance), to predict whether a customer will buy a bike. For more information about using the Naive Bayes Viewer, see Browse a Model Using the Naive Bayes Viewer. Data Required for Naive Bayes Models when we prepare data for use in training a Naive Bayes model, we should understand the requirements for the algorithm, including how much data is needed, and how the data is used.

IV. RESULT & DISCUSSION

Android is an open source software stack for mobile devices that includes an operating system, an application framework, and core applications. The operating system relies on a kernel derived from Linux. The application framework uses the Dalvik Virtual Machine. Applications are written in Java using the Android SDK, compiled into Dalvik Executable files, and packaged into .apk (Android package) archives for installation. The app store hosted by Google is called Google Play. In order to submit apps to Google Play, an Android developer first needs to obtain a publisher account. After submission, each .apk file gets an entry on the market in the form of a webpage, accessible to users through either the Google Play homepage or the search interface. This webpage contains meta-information that keeps track of information pertaining to the app (e.g., name, category, version, size, prices) and its usage statistics (e.g., rating, number of installs, user reviews). This information is used by users when they are deciding to install a new app. The Android system's built-in defense against malware consists of two parts: sandboxing each app and warning the user about the permissions that the app is requesting.

4.1 Data Cleansing

In the two market data sets, Market2011 and Market2012, we have observed the presence of thousands of apps that have similar characteristics. This kind of "duplication" can occur due to the following reasons:

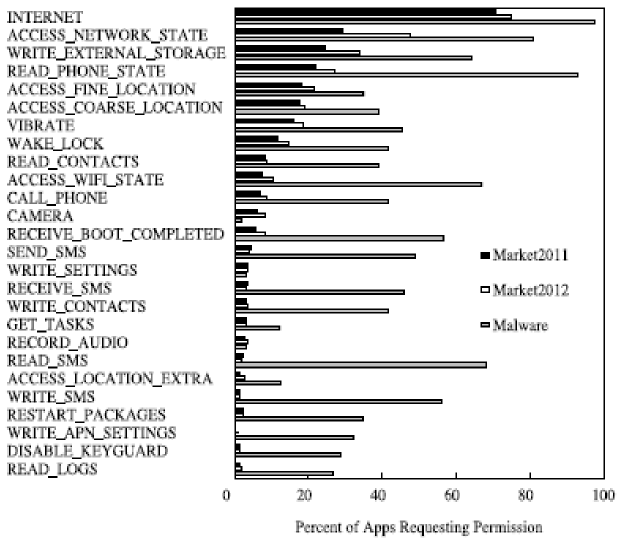
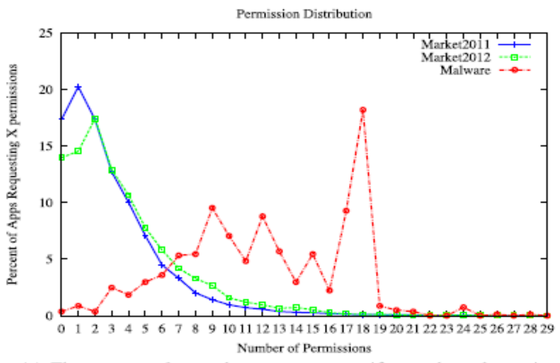


Fig 3; Kinds of Duplication

Slight variations (R1). one developer may release hundreds or even thousands of nearly identical apps that provide the same functionality with slight variation. A few examples include wallpaper apps, city or country specific travel apps, weather apps, or themed apps (i.e., a new app with essentially the same functionalities can be written for any celebrity, interest group, etc). _ App maker tools (R2). There are a number of tools that enable non-programmers to create Android apps. Often times many apps that are generated by these tools have similar app names and the same set of permissions.

4.2 Data Discussion

The frequencies of the most frequently requested permissions in the three data sets are presented in Fig.4. 1. There are 26 permissions in this figure, which include the top 20 for all three data sets. Clearly, for every permission, the percentage of malware apps is significantly higher than those in the two market data sets. For some permission, the difference is quite striking. For example, READ_SMS is requested by 67.95 percent of the malicious apps, but only 2.34 percent from Market2011, and 1.99 percent from Market2012. Furthermore, for almost every permission, a higher percent of apps in Market2012 request the permission when compared to the Market2011 data set. This shows a trend that proportionally more applications are requesting sensitive permissions. The exceptions to this are all related to SMS, where Market2012 actually saw a slight decrease for all permissions related to SMS. Fig. 2a shows the percentages of apps that request different numbers of permissions in the three data sets. From this graph, we again observe that malicious apps are requesting significantly more permissions than apps in the market data sets. However, there are many market data set apps that request much permission as well. Between Market2011 and Market2012, we also see a confirmation that apps are requesting a greater number of permissions on average. With proportionally fewer apps requesting zero or one permissions in Market2012. Overall, this indicates that the malicious apps are requesting permissions in different ways from normal apps, and indicates that looking at permission information is in fact useful. It also shows that there may be a slow evolution in the market data set. Interestingly, apps in the overlap data set, which are the "longer-lived" and more stable, generally request fewer permissions than other apps that are entering and exiting the app store or changing the permissions they are requesting.



(a) The percent of apps that request a specific number of permissions for each dataset.

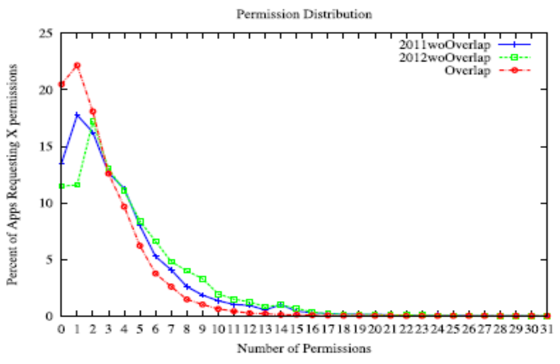
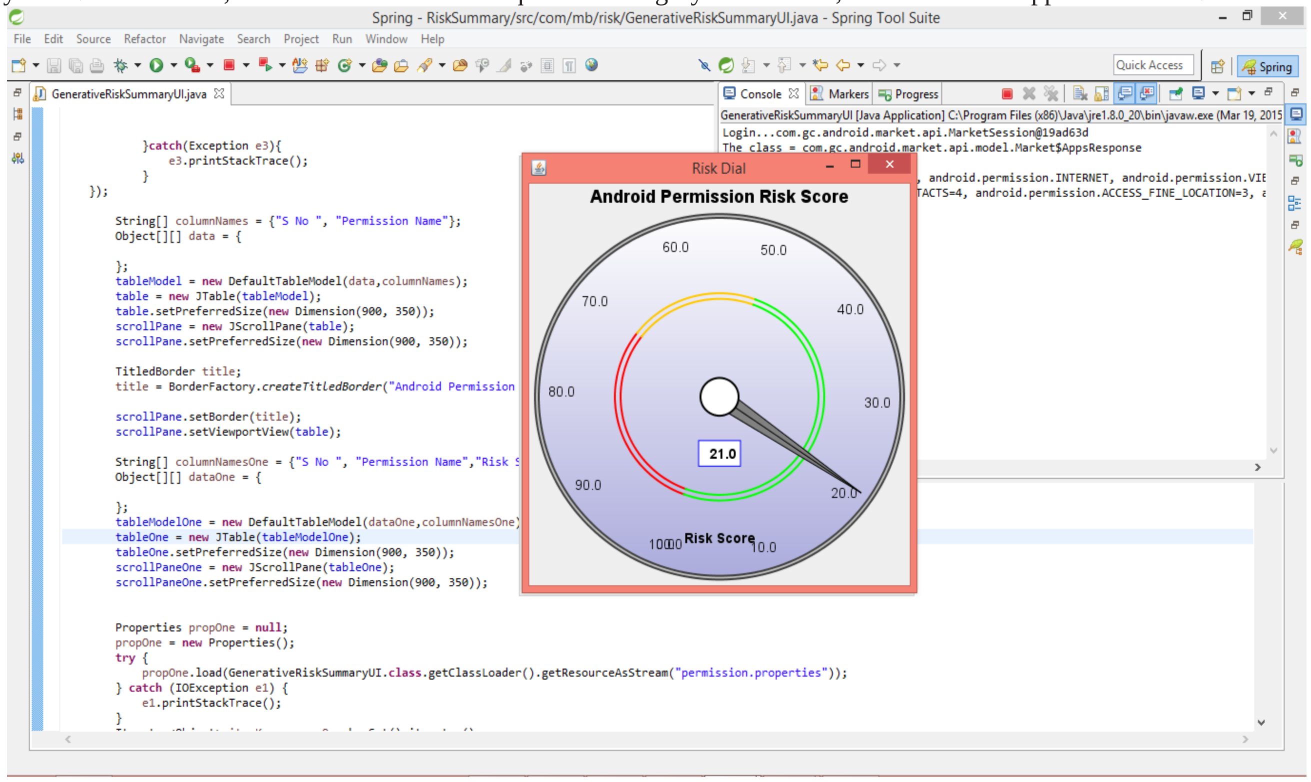


Fig 4: Number of Permission Distribution

4.3 Project Outcome

We aim at understanding how well the different techniques satisfy the second desideratum, namely, able to assigning high risks to known malware apps, and compare them to results from Section 4 as well as methods in the literature. Methodology. Most of our experiments are conducted with the 2011 data set with 10 fold cross validation.We divide the 2011 data set randomly into 10 groups. In each of the 10 rounds, we choose one different group as the test data set, and the remaining nine groups as the training data set. The models or statistics are trained on the training set, the generated model/statistic is used to compute the probabilities/ scores of apps in the testing set and the malware data set, and rank them together. When reporting the results, we use ROC curves, which plot the detection rate against warning rate if one chooses a particular risk value as indicative of malicious app. We also compute Area under Curve (AUC), which quantifies the quality of the ROC curves for a method. Here, AUC is the probability that a randomly selected malicious application will have a higher risk score than a randomly selected benign application. AUC values, however, can be misleading.

Knowing what percentage of malicious apps are among the top x percent high-risk apps for x > 20 is of little interest, since there are so many benign apps that have similar risks. This information, however, is more useful for smaller x values. We therefore also compute AUC for the X-axis (warning rate) of up to 5 and 10 percent, and use AUC5 and AUC10 to denote them. They evaluate what percentage of malicious apps will be rated among the apps with the highest risks under a ranking function. Parameter selection. For the Bayesian methods, both MNB and HMNB can be used with different number of hidden topics. From our experiments, we find that the maximum AUC for MNB is achieved with five hidden classes. And the maximum for HMNB is achieved with 80 hidden classes. We use these parameters when comparing with other methods. For a more complete discussion of the parameter selection, including graphs, we refer you to . Furthermore, both MNB and HMNB require the category information, which malware apps do not have.


Spring - RiskSummary/src/com/mb/risk/GenerativeRiskSummaryUI.java - Spring Tool Suite
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Spring
GenerativeRiskSummaryUI.java
Console
Markers
Progress
Risk Dial
Android Permission Risk Score
60.0
50.0
70.0
40.0
80.0
30.0
90.0
20.0
100.0
Risk Score
10.0
21.0

```
            }catch(Exception e3){
                e3.printStackTrace();
            }
    });

        String[] columnNames = {"S No ", "Permission Name"};
        Object[][] data = {

        };
        tableModel = new DefaultTableModel(data,columnNames);
        table = new JTable(tableModel);
        table.setPreferredSize(new Dimension(900, 350));
        scrollPane = new JScrollPane(table);
        scrollPane.setPreferredSize(new Dimension(900, 350));

        TitledBorder title;
        title = BorderFactory.createTitledBorder("Android Permission

        scrollPane.setBorder(title);
        scrollPane.setViewportView(table);

        String[] columnNamesOne = {"S No ", "Permission Name","Risk S
        Object[][] dataOne = {

        };
        tableModelOne = new DefaultTableModel(dataOne,columnNamesOne)
        tableOne = new JTable(tableModelOne);
        tableOne.setPreferredSize(new Dimension(900, 350));
        scrollPaneOne = new JScrollPane(tableOne);
        scrollPaneOne.setPreferredSize(new Dimension(900, 350));


        Properties propOne = null;
        propOne = new Properties();
        try {
            propOne.load(GenerativeRiskSummaryUI.class.getClassLoader().getResourceAsStream("permission.properties"));
        } catch (IOException e1) {
            e1.printStackTrace();
        }
```

```
GenerativeRiskSummaryUI [Java Application] C:\Program Files (x86)\Java\jre1.8.0_20\bin\javaw.exe (Mar 19, 2015
Login...com.gc.android.market.api.MarketSession@19ad63d
The class = com.gc.android.market.api.model.Market$AppsResponse

, android.permission.INTERNET, android.permission.VIE
TACTS=4, android.permission.ACCESS_FINE_LOCATION=3, a
```

Fig 5: Sample Sreen shot

When assigning a category to malware apps, we use a method that gives the most advantage to malware. More specifically, we compute the probability of each malicious app for every category and assigns the app to the category in which the app has the highest probability.

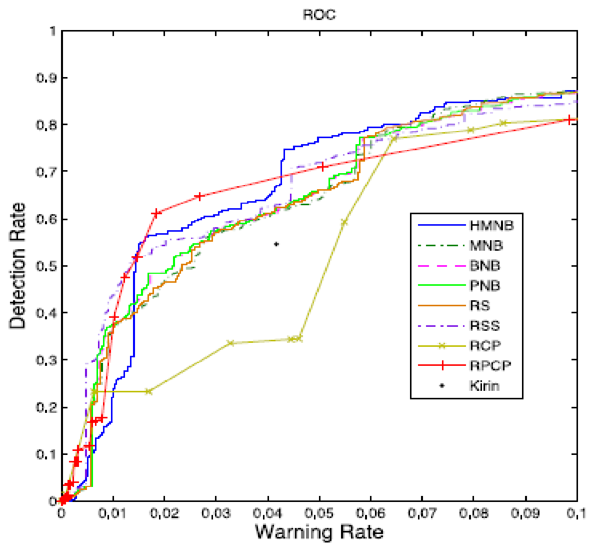


Fig 5.1: Finding warning rate

V. CONCLUSION

We discuss the importance of effectively communicating the risk of an application to users, and propose several methods to rate this risk. We test these methods on large real-world data sets to understand each method's ability to assign risk to applications. One effective method is the RSS method which has several advantages. It is monotonic, and can provide feedback as to why risk is high for a specific app and how a developer could reduce that risk. It performs well in identifying most current malware apps as high risk. This method allows for highly critical permissions and less-critical permissions to affect the overall score in an easy to understand way, making it more intuitive as well as difficult to evade when compared with other models.Enhanced Naive Bayes has been extensively used both in the context of spam detection and anomaly detection in network traffic flows. In the context of Android, however, there has been limited work. Its presents a behavioral-based detection framework for Android that realizes a host-based intrusion detection system that monitors events originating from the device and classifies them as normal or abnormal. Our work differs in that we use machine learning for the purpose of risk communication.

REFERENCES

[1] S. Chakradeo, B. Reaves, P. Traynor, and W. Enck, "MAST: Triage for Market-Scale Mobile Malware Analysis," Proc. Sixth ACM Conf. Security and Privacy in Wireless and Mobile Networks (WISEC'13), pp. 13-24, 2013

[2] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution,"Proc. 33rd IEEE Symp. Security and Privacy, May 2012.

[3] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen, "Investigating User Privacy in Android Ad Libraries,"Proc. IEEE Mobile Security Technologies (MoST '12), 2012.

[4] B.P. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, "Android Permissions: A Perspective Combining Risks and Benefits,"Proc. 17th ACM Symp. Access Control Models and Technologies (SACMAT '12), 2012.

[5] R. Potharaju, A. Newell, C. Nita-Rotaru, and X. Zhang, "Plagiarizing Smartphone Applications: Attack Strategies and Defense,"Proc. Fourth Int'l Conf. Eng. Secure Software and Systems, pp. 106-120, 2012.

[6] H. Peng, C. Gates, B. Sarma, N. Li, Y. Qi, R. Potharaju, C. NitaRotaru, and I. Molloy, "Using Probabilistic Generative Models for Ranking Risks of Android Apps,"Proc. Conf. Computer and Comm. Security, (CCS '12), pp. 241-252, 2012.

[7] P.G. Kelley, S. Consolvo, L.F. Cranor, J. Jung, N. Sadeh, and D. Wetherall, "A Conundrum of Permissions: Installing Applications on an Android Smartphone," Proc. Workshop Usable Security (USEC '12), Feb. 2012.

[8] M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, "RiskRanker: Scalable and Accurate Zero-Day Android Malware Detection," Proc. 10th Int'l Conf. Mobile Systems, Applications, and Services, (MobiSys '12), pp. 281-294, 2012.

[9] A.P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner, "Android Permissions: User Attention, Comprehension, and Behavior,"Proc. Eighth Symp. Usable Privacy and Security, article 3, 2012

[10] K.W.Y. Au, Y.F. Zhou, Z. Huang, and D. Lie, "PScout: Analyzing the Android Permission Specification,"Proc. ACM Conf. Computer and Comm. Security (CCS '12), pp. 217-228, 2012.

[11] P.G. Kelley, S. Consolvo, L.F. Cranor, J. Jung, N. Sadeh, and D.Wetherall, "A Conundrum of Permissions: Installing Applications on an Android Smartphone," Proc. Workshop Usable Security (USEC '12), Feb. 2012.

[12] P.G. Kelley, L.F. Cranor, and N. Sadeh, "Privacy as Part of the App Decision-Making Process," Proc. Conf. Human Factors in Computing Systems (CHI '13), pp. 3393-3402, 2013.

[13] T. H.-J. Kim, P. Gupta, J. Han, E. Owusu, J. Hong, A. Perrig, and D. Gao, "OTO: Online Trust Oracle for User-Centric Trust Establishment," Proc. ACM Conf. Computer and Comm. Security, pp. 391- 403, 2012.

[14] J. Lin, S. Amini, J.I. Hong, N. Sadeh, J. Lindqvist, and J. Zhang,"Expectation and Purpose: Understanding Users' Mental Modelsof Mobile App Privacy Through Crowdsourcing," Proc. ACM Conf. Ubiquitous Computing (UbiComp '12), pp. 501-510, 2012.

[15] R.D. Luce, Response Times: Their Role in Inferring Elementary Mental Organization. Oxford Univ. Press, 1986.

[16] S. Mishra, M. Gregson, and M.L. Lalumi_ere, "Framing Effects and Risk-Sensitive Decision Making," British J. Psychology, vol. 103, no. 1, pp. 83-97, Feb. 2012.

[17] S. Motiee, K. Hawkey, and K. Beznosov, "Do Windows Users Follow the Principle of Least Privilege?: Investigating User Account Control Practices," Proc. Sixth Symp. Usable Privacy and Security, 2010.

[18] H. Peng, C.S. Gates, B.P. Sarma, N. Li, Y. Qi, R. Potharaju, C. Nita- Rotaru, and I. Molloy, "Using Probabilistic Generative Models for Ranking Risks of Android Apps," Proc. ACM Conf. Computer and Comm. Security, pp. 241-252, 2012.

[19] E.E. Schultz, "Web Security, Privacy, and Usability," Handbook of Human Factors in Web Design, K.-P.L. Vu and R.W. Proctor, eds., pp. 663-677, CRC Press, 2011.

[20] J. Schwarz and M. Morris, "Augmenting Web Pages and Search Results to Support Credibility Assessment," Proc. SIGCHI Conf. Human Factors in Computing Systems, pp. 1245-1254, 2011.

[21] J. Staddon, D. Huffaker, L. Brown, and A. Sedley, "Are Privacy Concerns a Turn-Off?: Engagement and Privacy in Social Networks," Proc. Eighth Symp. Usable Privacy and Security (SOUPS '12), pp. 1-13, 2012.

[22] S. Sternberg, "Inferring Mental Operations from Reaction-Time Data: How We Compare Objects," An Invitation to Cognitive Science: Methods, Models and Conceptual Results, D. Scarborough and S. Sternberg, eds., pp. 703- 863, MIT Press, 1998.

[23] J. Sun, P. Ahluwalia, and K.S. Koong, "The More Secure the Better? A Study of Information Security Readiness," Industrial Management and Data Systems, vol. 111, no. 4, pp. 570-588, 2011.

[24] A. Tversky and D. Kahneman, "The Framing of Decisions and the Psychology of Choice," Science, vol. 211, no. 4481, pp. 453-458,1981.

[25] W. Van Wassenhove, K. Dressel, A. Perazzini, and G. Ru, "A Comparative Study of Stakeholder Risk Perception and Risk Communication in Europe: A Bovine Spongiform Encephalopathy Case Study," J. Risk Research, vol. 15, no. 6, pp. 565-582, 2012.

AUTHORS

S. Santhi doing the degree of **Master of Philosophy** in computer science from **Shri Sakthikailaash Womens College, Salem, Tamilnadu**. The research interest includes naive bayes, binary risk signal, data mining.