Permission based Malware Analysis & Detection in Android

Ridhima Seth
M.Tech. (Information Security Management)
Indira Gandhi Delhi Technical University for Women
Kashmere Gate, Delhi, India
+91-858901654
ridhima90@gmail.com

Rishabh Kaushal
Asst. Prof., Department of Information Technology
Indira Gandhi Delhi Technical University for Women
Kashmere Gate, Delhi, India
+91-9810774114
rishabh.kaushal@gmail.com

ABSTRACT
Android being a leading and the most popular operating system for smart phones and tablets, has also become a prime target for the attackers due to its growing users and it being an open source platform. This document describes the work done in detecting malware in the Android platform by performing static analysis on the permission based framework in Android platform. In our work, we have extracted a number of permission based features by disassembling the Android application (apk) files. Features, thus extracted, have been assembled in feature vectors. Machine learning algorithms are then applied to these feature vectors in order to detect malicious applications. Three algorithms, namely, K Nearest Neighbors (KNN), Decision Tree (DT) and SVM are applied on an initial dataset of 90 applications containing 55 malicious and 35 benign which is being currently expanded. Out of the three learning algorithms, Decision Tree (DT) is giving the best results, with classification accuracy of 96%.

Keywords
Android Malware, Static analysis, Machine Learning, Permission Framework

1. MOTIVATION
There has been a whopping increase in the Android applications in the past few years, which has also given attackers an opportunity to steal and misuse sensitive information of the users or charge them certain amount unknowingly. Android operating system exhibits robust security architecture and provides security at various levels of its layered architecture. The Permission framework of Android platform is one of the important features for providing access controls. Each application is granted a set of permissions, which control its access to certain privacy sensitive resources. The application can have access to these privacy sensitive resources only when the permissions that the application asks for have been granted by the user at the time of installation of the application. However these permissions are usually ignored by a normal user when he is installing the application, which pose as a risk. Our work therefore analyses a set of potentially dangerous permissions and detects presence of malware on that basis.

2. METHODOLOGY
Our current work focuses on detection of Malware using static analysis techniques. Static analysis [1][2][3][4] refers to analyzing the source code for malicious patterns without actually running the code. An important static analysis technique focuses on analyzing the manifest file (AndroidManifest.XML) included in every application for the set of permissions used and other components like Services, Broadcast receivers and Intents. As a first step each application is disassembled using a command line tool called Apktool [6] to generate the manifest file and the source code. Then the permissions from the manifest file are extracted using an XML parser written in Java. A feature vector for each application is created based on a total of 35 permissions which is considered as the feature set. The accuracy of detecting the malware samples based on the features extracted is evaluated by applying classification algorithms like K Nearest Neighbors, Decision Tree and SVM.

2.1 Dataset
A total of 55 malicious applications and 35 benign applications have been collected manually. The following datasets have been used to collect malicious application.

2.1.1 Android Malware Genome Project
This dataset consists of over 1200 Android applications containing malware samples which cover majority of Android malware families. This dataset was released by Zhou et al. [5] as a part of their research work.

2.1.2 Contagio Dump
This is a publically available collection of Malware samples. [7]
3. RESULTS
After the feature vectors are created, the accuracy of malware detection has been calculated by applying three supervised machine learning algorithm namely K Nearest Neighbors, Decision Tree and SVM. Out of the 90 applications, 60 applications were used as the training data and 30 as test data. In order to evaluate the efficiency of the method used, a confusion matrix is constructed to measure the accuracy of the experiments. The matrix uses the following metrics:

\[ \text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \]

KNN was applied with different values of k, and the highest accuracy obtained by the classifier was 80% with k = 3. The best results were obtained by using the Decision Tree classifier with a correct classification accuracy of 93.33%. Table 1-3 give the confusion matrices obtained. Figure 2 gives a comparison of the three classifiers.

Table 1. Confusion Matrix for K Nearest Neighbors

<table>
<thead>
<tr>
<th></th>
<th>Benign</th>
<th>Malware</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Benign</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>Malware</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 2. Confusion Matrix for Decision Tree

<table>
<thead>
<tr>
<th>Decision Tree</th>
<th>Benign</th>
<th>Malware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benign</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>Malware</td>
<td>1</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 3. Confusion Matrix for Support Vector Machine

<table>
<thead>
<tr>
<th>SVM</th>
<th>Benign</th>
<th>Malware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benign</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>Malware</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

A comparison between certain permission in both Malware and benign applications is shown in Figure 3.

4. IMPLICATION OF RESULTS
This work is a first step of our ongoing efforts towards building an effective web based malware analysis tool by improving upon the previous research. The result of this work achieves limited success and indicates that static analysis alone by analyzing the manifest file for permissions is not enough. Also efficient malware detection techniques in the presence of various attacks like transformation attacks, mimicry and poisoning attacks need to be explored.

5. REFERENCES


