Project Title: Predicting Vulnerable Risk Score of Android Apps From Software Metrics
Team Name: Repo Miners

Introduction

With the increase in popularity and use of Android apps, security and privacy issues of Android apps are becoming a concern [1, 3]. Android app developers can benefit in writing better quality apps if a study can systematically analyze which software engineering issues such as coding conventions and software metric contribute to Android app vulnerability. Prior studies have shown how software related metrics can be used to predict software defects. We draw inspiration from these studies and identify the possibility of applying data mining techniques to predict vulnerability in Android apps. In this project we aim to predict vulnerability risk score for each version of apps from different features from the dataset such as number of commits, number of contributors, number of classes, number of lines per code, and number of functions.

Dataset

The dataset [2] is a 87. 5 MB SQLite database containing software related metrics data and vulnerability data of 1179 open source Android apps that has 4416 different versions. The dataset contains a vulnerability risk score for each of the versions of the 1179 applications. The dataset also contains software metrics for each version of the 1179 apps such as number of files, number of directories, file complexity, and number of violations.

Implementation

For systematic access of the dataset we have used the sqlite3 Python library. We have written code to access this database, and can access the values of the dataset. As next step, we want to identify interesting features that can help in predicting vulnerability scores. For this step we plan to perform correlation (Pearson, Spearman, MIC) using Python libraries. The most correlating features will be used to predict the app vulnerability score. For predicting vulnerability risk score we plan to use the correlating features as inputs to standard machine learning algorithms such as naive bayes, support vector machines, and decision trees. We plan to use the SciKit library available in Python. To evaluate the performance of classifier we plan to use cross-validation methods available in SciKit.

References

1. Long Lu, Zhichun Li, Zhenyu Wu, Wenke Lee, and Guofei Jiang. 2012. CHEX: statically vetting Android apps for component hijacking vulnerabilities. In *Proceedings of the 2012 ACM conference on Computer and communications security* (CCS '12). ACM, New York, NY, USA, 229-240.
2. Daniel E. Krutz, Mehdi Mirakhorli, Samuel A. Malachowsky, Andres Ruiz, Jacob Peterson, Andrew Filipski, and Jared Smith. 2015. A dataset of open-source Android applications. In *Proceedings of the 12th Working Conference on Mining Software Repositories* (MSR '15). IEEE Press, Piscataway, NJ, USA, 522-525.
3. http://www.scmagazine.com/researcher-discovers-thousands-of-vulnerable-apps/article/410418/

(1) Not clear which prediction technique you are going to implement?
(2) For feature selection, look at weka ml system first
(3) NB, SVM, and DT are classifiers; if your objective is prediction, then look at regression, neural networks, Gaussian process learning, etc.