**Malware Detection: STA 221 Final Project**

**Jasper Tsai, Matthew Chen, Mark Faynboym**

**Abstract:**

As of 2023, Android has a global market share of 71.8% in the mobile operating systems market. The rapid growth of Android apps and its popularity in the smartphone market has made it an easy and accessible target for malware. In the Journal of Information Security and Applications, Mathur et. al. explored the strategy of using native AOS permissions and developers’ custom permissions to detect malware with the use of classification methods. In this project we attempt to replicate the results found in Mathur’s NATICUSdroid article as well as attempt to further the performance of detection, which stands at a 97% accuracy with an FPT rate of 3.32% and F-measure of 0.96.

**Introduction/Background**

In the past decade, the Android operating system has gained immense worldwide popularity and as a result has been the target of malware due to its open-source nature. Although Android attempts to curb the harm of malicious software with updates to fix vulnerabilities, malware apps have also adapted. The newly created native AOS as well as custom permissions could be of great importance in detecting current malware as older research circa 2010-2012 involved permissions that are not widely used or do not exist anymore.

The reason why malware is significant is that it compromises the security of a device, using its resources, or accessing private data. Google’s Play Store has so far helped in detecting 700,000 malicious apps. However many of which were not detected. A significant portion of undetected malware were found responsible for DDoS attacks, stalkerware apps with downloads caught accessing user data without consent and those pretending to be gaming or photography apps which would only later be detected and removed. Android apps aside, Samsung, Huawei and other popular third party app stores are also prone to malware infection. This poses a significant threat to users’ privacy and data when they download apps from such third-party app stores.

Android’s API level allows for interactions with it’s AOS to utilize a permission-based model which allows users to control an app’s access to device resources. The significance of identifying malware using ML specifically in using custom and native permissions together have not been much considered before Mathur’s article, and those that have, used outdates permissions going back 10 years. The further exploration of these methods could lead to significant increases in cybersecurity of open-source operating systems for mobile devices.

**Exploratory Data Analysis**

Source: Our data source comes from the UC Irvine Machine Learning Repository under the title ‘NATICUSdroid Dataset’. The dataset contains permissions extracted from 29,333 benign and malware Android apps released between 2010 and 2019. 86 features with one target column and 14,700 were found to be malware and the other 14,632 found to be benign. The UCI Repository dataset was derived in two parts the benign dataset and the malware dataset. The benign app dataset came from Androzoo

which is a large collection of Android apps from several sources including Google Play Store. Although locations like the Google Play Store are known to contain malware, for simplicity, an assumption was made that the apps from these sources exhibit non-malicious behavior. To add additional assurance the 15,000 apps chosen were rated ‘benign’ by VirusTotal and then further pruned using a threshold of a specific API protection level. To create the malware, the Argus Lab’s Android Malware Database was used that contained 24,500 malware from 2010-2019, 14,700 of which were sampled. The sample size was chosen to be roughly the same amount as the benign dataset.

**Methodology**

**Results**

**Interpretations**

**Discussion**

Conclude your analysis in this section. You can touch on the following topics.

* A brief recap of this project.
* Findings in the methodology
* Suggestions for future research and/or policy making given your findings.
* Caveats of the current analysis.