Cam Spotter

David Oyeka, Hana DeClerck, Jesus Contreras

Android Security Analysis

Introduction

Globally, Android devices are the most widely used among mobile devices. With such a large user-base, we wanted to determine how secure the Android operating system is.

Hypothesis

1)The frequency of malware attacks on Android devices increases due to newly developed threats that bypass Android security protocols.
2)The more permissions an app has, the more likely it is to be malicious.

Data

- Android potentially harmful application install rates
- Android permissions ML dataset

Testing Methods

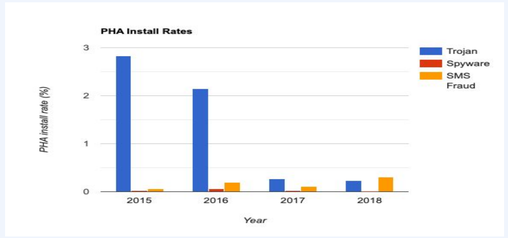
- Linear regression
  - Analyzing these coefficients
- Explore yearly trends of top 3 PHAs (Trojan, SMS fraud, spyware)

Results

- From the yearly PHA data, it can be interpreted that the rate of PHAs being installed on Android devices decreases from the years 2014-2018. There appears to be an overall decreasing trend in the efficacy of these PHAs' downloads onto Android devices.

- We ran a multiple regression model to show the relationship between 114 variables and the whether or not an app was malicious. Of the 114 variables, 113 are permissions and the last is the number of permissions. To the right, we have a table that displays the coefficients of the number of permissions and the 5 variables with the highest significance. We found that the coefficient of number of permissions was not significant, having a value of .002. According to our data, the variables that have the greatest effect on whether or not an app is malicious are allowing the ability to send messages(coeff=.441), allowing the ability to delete cache files(coeff=.330), and allowing the ability to track the current location of the device(coeff=.276).

Graphs & Tables



Multiple Regression		
Variable	**Coefficient**	**P-Value**
SEND_SMS (1=permissions granted)	.441	.000
DELETE_CACHE_FILES (1=permissions granted)	.330	.000
CONTROL_LOCATION_UPDATES (1=permissions granted)	.276	.000
SET_ACTIVITY_WATCHER (1=permissions granted)	-.274	.002
ACCESS_LOCATION_EXTRA_COMMANDS (1=permissions granted)	.262	.000
# OF PERMISSIONS	.002	.030

Conclusion

From our data collection and testing, we fail to accept our hypothesis that Android devices are more susceptible to attacks due to an increasing ecosystem of malware applications. From our linear regression results, there are no correlations between the number of permissions an app has and whether or not the application is compromised. Our results, however, indicate that certain permissions are related to whether or not an app is malicious. Further exploration is required to find out if there is a relationship between specific permissions and an app's intent.

Works Cited

http://darwin.rit.edu/reports,
https://pages.nokia.com/T003B6-Threat-Intelligence-Report-2019.html,
https://securelist.com/mobile-malware-evolution-2019/96280/,
https://docs.greynoise.io/#greynoise-api,
https://source.android.com/security/reports/Google_Android_Security_2018_Report_Final.pdf,
https://figshare.com/articles/Android_malware_dataset_for_machine_learning_2/5854653

Capstone: Computer Vision Camera Detection

Motivation

We wanted to further explore mobile security and privacy by using a neural network to identify when a third party device might be taking a photo of your phone screen. We hypothesize that we can achieve 70% accuracy on this prediction task.

Data

- Images of: phones, people taking pictures of phones, phones taking pictures, etc.
- Scraped from Google Images & generated with iPhone bursts
- More images were generated via reflection, and slight rotations

Potential Cause for Model Error

- Non-realistic scraped images
- Almost all images were modified from their original format to fit the model

Testing Methods

- Convolutional Neural Network (CNN)
- 80/20 train-test split on the data

Results

The images were labelled according to the prompt: "Is there a smart phone in a position to take a picture in this image?" For reference, 73% of our data was labeled yes, while 27% of our data was labelled no.

- If the network always chose 'yes', the testing accuracy was 73%.
- If the network always chose 'no', the testing accuracy was 27%.
- If the network trained by guessing randomly, the test prediction accuracy was 49%.
- If the network trained normally, the test prediction accuracy was 64%.

Example Images

Conclusion

Given the results of our image model, we would fail to accept our image hypothesis. Intuitively, we were disappointed by the results, especially given that we could simply guess yes and achieve the best accuracy. This is most likely due to the limited dataset of images, as it was difficult to locate a lot of data characterizing unique phones.

Works Cited

https://github.com/Joeclinton1/google-images-download

Acknowledgements: Prof. Ellie Pavlick, HTA Sol Zitter

CS1951a Final Project -- Spring 2020