SPrivacy: Context dependent user data privacy in Android

Prajit Kumar Das*, Bin Liu†, Anupam Joshi*, Tim Finin* and Hongxia Jin†
*University of Maryland Baltimore County
Email: prajit1,joshi,finin@umbc.edu
†Samsung Research America
Email: bin2.liu,hongxia.jin@samsung.com

***Abstract*—Some abstract R: review. The structure should be: 1) motivation, 2) what has been done before, 3) goal (context enrichment), 4) method (ontology, reasoner, sharing, inconsistency checking)**

I. Introduction

Android Privacy is a big challenge. This paper is our attempt at solving it. [1]

Broadly these will be the sections in the paper but we might change this later:

1) Introduction
2) Overview
3) System Design and Architecture
4) Implementation Details
5) Experimental Evaluations
6) Discussion and Related Work
7) Conclusion

II. Overview

...

Android mobile devices use Content Providers in order to allow applications (or apps in short), installed on the device, access to user data. Android's permission mechanism controls whether such an access will be allowed or not. However, the current permission model for Android is a "all or nothing" model. The user has to either accept all the permissions that the app requests or they cannot install the app. Such a model is overtly restrictive and does not allow the possibility of execution time permission granting. It also does not allow the user to restrict the data or allow data access based on contextual situations. Current generation of mobile devices are capable of collecting a lot of user information using in-built sensors or by accessing user personal contacts, calendar, emails and messages. In such a situation it has become necessary that user have more control over their data. There needs to be a better way of controlling such data. In this paper we implement a new privacy aware middle-ware, SPrivacy. SPrivacy allows the users to dynamically control the data that an app is allowed to access.

Traditional solution(s) for controlling user data privacy on mobile devices have focused on two techniques:

1) Make changes to the mobile operating system and control how data flows from the creator to the consumer of the data on the device. This is done by creating custom ROMs or mobile operating systems and modifying the APIs in the custom operating system to achieve the goal mentioned before.
2) Obtain elevated privileges from the operating system in order to control how the data flows. In this mechanism an app is installed on the device with "root" privileges on the device so that the app is able to modify the behavior operating system APIs as per its needs.

We argue that both these techniques potentially create loopholes which might be used by a malicious app towards its own untoward actions. Our proposed mechanism allows us to have no changes on the operating systems but at the same time allows us to control the data that an app can access on the device. We achieve this by implementing SPrivacy which is a middle-ware that is capable, based on a list of settings and a URI redirection mechanism, of controlling the data app(s) get. In our mechanism we take apps from the Android app market and modify the Content Provider URIs they use in order to ensure they call our middle-ware instead. Our middle-ware on then determines if the app will be given access to the data or not.

III. System Design and Architecture

system design

IV. Implementation Details

...

Taming Information-Stealing Smartphone Applications (on Android) - Our closest competitor. However, they use only 24 apps to analyze their data. Which is odd and they use 13 "known to be rogue apps" based on TaintDroid system's analysis. The rogue apps leak information which are sensitive like IMEI number and location information. They do not expalin why they just use 24 apps though.

V. Experimental Evaluations

...

Taming Information-Stealing Smartphone Applications (on Android) - Our closest competitor. However, they use only 24 apps to analyze their data. Which is odd and they use 13 "known to be rogue apps" based on TaintDroid system's analysis. The rogue apps leak information which are sensitive

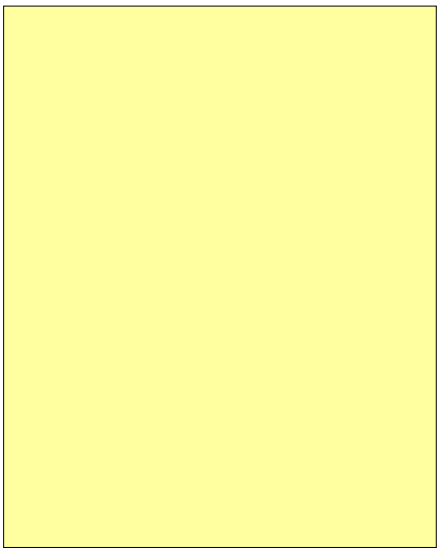

Fig. 1. High-level architecture of our system.

like IMEI number and location information. They do not expalin why they just use 24 apps though.

VI. Related Work

...

Taming Information-Stealing Smartphone Applications (on Android) - Our closest competitor. However, they use only 24 apps to analyze their data. Which is odd and they use 13 "known to be rogue apps" based on TaintDroid system's analysis. The rogue apps leak information which are sensitive like IMEI number and location information. They do not expalin why they just use 24 apps though.

VII. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] R. Amadeo, "App ops: Android 4.3's hidden app permission manager, control permissions for individual apps!" July 2013. [Online]. Available: http://goo.gl/w9CjrQ