An Analysis of Pre-installed Android Software

Julien Gamba[†◇], Mohammed Rashed[◇], Abbas Razaghpanah[‡], Narseo Vallina-Rodriguez[†∗], Juan Tapiador[◇]
† IMDEA Networks Institute, ◇ Universidad Carlos III de Madrid, ‡ Stony Brook University, ∗ ICSI

***Abstract*—Thanks to the openness of the Android ecosystem, mobile device vendors can build and sell smart phones and other mobile devices using their own custom versions of Android. Most of these custom versions deviate significantly from Google's official Android Open Source Project (AOSP): in addition to various visual and functional changes to the base OS, vendors add proprietary applications (apps hereafter) to their firmware, and sometimes even add custom (often unknown) certificates to the system's root certificate store. In fact, recent anecdotal evidence has revealed that pre-installed apps can put, intentionally or not, user's privacy and security at risk. This is especially concerning for lesser-known brands producing lower-end devices for whom preserving user privacy might not be high on the priority list. In this extended abstract, we present our methodology to explore the complex and diverse ecosystem of Android pre-installed apps as well as our preliminary results.**

***Index Terms*—Android; security; privacy; measurements**

Tipo de contribución: *Investigación en desarrollo*

I. Introduction

A modern Android phone comes with multiple apps and services pre-installed. The openness of the Android source code makes it possible for any manufacturer to ship a custom version of Android, along with some proprietary pre-installed apps on the system partition. These apps can be useful to users (*e.g.*, a browser or a calculator) but can also be unwanted. Moreover, pre-installed apps, specifically those installed in `/system/priv-app/` or signed with the platform signing key, may have access to privileged system permissions which are not available to user-installed apps. In many cases these apps run in the background, tracking user activities without their explicit consent, and often without their knowledge.

Some vendors have recently come under scrutiny by the media for these practices. For instance, it has been reported that OnePlus devices contain software that allows a remote controller to root the phone and perform other high-privilege operations that are reserved for the manufacturer [1]. Such modifications are typically introduced by manufacturers, but may also be done by network operators and phones resellers. So far, no research study has systematically studied the privacy and security risks of Android OS modifications beyond the addition of certificates in the trusted root store [2]. Consequently, pre-installed apps have remained a largely unknown area. It is unclear whether vendors use these apps only to harvest personal data from consumers, or provide APIs to affiliate apps and partners to access privileged resources, as in the case of the Samsung Knox API. While it is possible to avoid these potential abuses and vulnerabilities by installing more widely-trusted and open-source alternatives to the stock firmware (*e.g.*, LineageOS), it should be noted that it is far from an ideal solution as rooting devices exposes users to further security risks, it is neither easy nor accessible for most users to try, and that a third-party firmware can reduce the functionality of the device due to missing drivers and a slew of other issues.

This project, which is still in its early stages, seeks to shed light on the presence of pre-installed Android apps across vendors and devices, studying them in depth to answer the following questions:

- What is the ecosystem of pre-installed apps, including all actors in the supply chain?
- Do pre-installed apps leak personally identifiable information (PII)? If so, with whom do they share this information and for what purpose?
- Do such apps present security vulnerabilities?
- What are the relationships between vendors and the potential app developers for their pre-installed apps?

To that end, we are currently gathering a large corpus of pre-installed apps by crowd-sourcing them from real user devices. Once this is done, we will apply both static and dynamic analysis techniques to these binaries. Finally, we aim to identify the origins of those apps, hoping to attribute their development to third party app developers using signature matching techniques. In this extended abstract, we will first detail our methodology (Section II) and then present some preliminary results (Section III).

II. Datasets and methodology

Most of the pre-installed apps cannot be found on traditional app stores; instead, they must be extracted from the system partition from real phones. We called for volunteers and extracted over 1,000 unique pre-installed APKs out of 15 devices, covering 8 different manufacturers–including both high-end and low-end vendors like Samsung and Wiko, respectively. None of these APKs is listed on Google Play. We obtained written consent from all users before we harvested anything from their phone, even though no personal data was obtained.

Our second dataset comes from the Lumen Privacy Monitor app [3]. Lumen is a home-built Android app, publicly available on Google Play, that aims to promote mobile transparency and enable user control over their personal data and traffic. Lumen leverages the Android VPN permission to intercept and analyze all Android traffic on user-space and in-situ, even if encrypted. For this study, we use over 15M anonymized traffic logs provided by over 13,000 users from over 120 countries. This dataset covers 567 pre-installed apps[1] found in 140 different Android vendors. We reference the reader to Lumen's previous work [3] to get a better understanding of its capabilities and its mechanisms to generate accurate traffic fingerprints on an per-application basis.

III. Preliminary results

A. Static Analysis

We applied basic static analysis techniques to the apps in our dataset to analyze permission usage, access to privileged resources and privacy leaks.

PII Leaks: We decompiled and manually investigated APKs from our dataset to look for PII leaks. We found some apps leaking the IMEI of the phone through SMS along with other

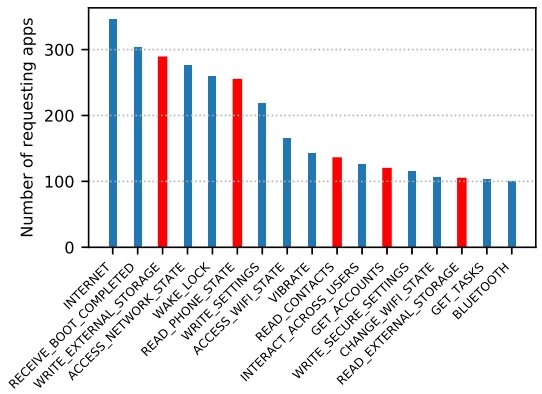

Figure 1: Most popular permissions among pre-installed apps. Red bars flag those permissions considered as "Dangerous" by Google [5].

Vendor	Custom Permissions
Samsung	188
Sony	138
Google	37
Meizu	15
HTC	13

Table I: Number of custom permissions by vendor

items of private information. We also found apps geo-targeting users from specific countries: first the pre-installed service checks the geographical location of the user and then, it leaks PII information if the user is located in a given country.

Android Permissions: The total number of permissions requested by pre-installed apps in our dataset is 1,064. Figure 1 shows the most requested permissions. The plot only shows the most popular permissions, *i.e.*, those requested by at least 100 apps. The red bars are the permissions flagged as "Dangerous" by Android [5] (*e.g.*, Location). As we can see, most pre-installed apps require Internet access, and a substantial amount of them can read the phone state, the list of contact and the accounts.

However, more than 99% of the permissions identified are custom permissions, defined by the app developer. Table I shows the number of custom permissions for some vendors in our dataset. These permissions are used for authentication mechanism, for billing or to control IoT devices. Samsung is the vendor with a largest number of custom permissions, including those associated with the Knox API. Google also offers additional permissions besides basic Android ones. For instance, `com.google.android.googleapps.permission.GOOGLE_AUTH` is used by Google services to authenticate the user with Google servers.

B. Lumen Analysis

Lumen detected personal data dissemination to cloud services emanating from 407 pre-installed apps, including 77 apps that where extracted from volunteers' phones. Note that not all of the apps from volunteers were put under Lumen's scrutiny: we expect to find more personal data dissemination by inspecting every app in our dataset.

[1]For this study, we consider a Lumen-analyzed app as pre-installed if it is not detected by the Androzoo project [4]. We are currently exploring more accurate methods to distinguish pre-installed software.

Domain	Percentage of PII leaks
data.flurry.com	1.24%
android.clients.google.com	1.18%
www.google.com	0.88%
pagead2.googlesyndication.com	0.87%
googleads.g.doubleclick.net	0.86%
settings.crashlytics.com	0.85%

Table II: Top third-party domains used by pre-installed apps

Traffic Analysis: We found 7,613 unique domains that receive data from pre-installed apps. Up to 79% of these communications are done over encrypted channels. Among these domains we found domains that are used to serve ads to the user (*e.g.*, Google's DoubleClick), and also other tracking and analytics services (*e.g.*, Crashlytics and Verizon's Flurry).

PII Leaks: We found that 63.2% of personal data is disseminated to first-party domains, and the remaining 36.8% to third-party domains. Table II shows the most popular third-party domains among the pre-installed apps in our dataset. Further, 45.3% of these flows containing personal data are sent to third-party domains. We have observed that pre-installed services may upload sensitive data without encryption, and therefore constuting a serious privacy risk for users.

IV. Conclusion and Ongoing Work

To the best of our knowledge, our work is the first study of pre-installed apps and services in Android devices. The size and disparity of the Android ecosystem makes a thorough study challenging but crucial for users' security and right to privacy. Even with a limited size dataset, we managed to find numerous PII leaks and bad practices in pre-installed apps.

We are extending our dataset by scaling up our crowdsourcing campaign. Then, we'll leverage FlowDroid [6]–a static taint analysis tool for Android apps — to study the behaviour of our apps more deeply. We are also complement our analysis using the mobile cyber-intelligence dataset provided by Eleven Paths' Tacyt which contains millions of APKs [7] to identify relationships between pre-installed services and publicly available apps. Tacyt will allow us to cross-match, for instance, a given URL or custom permission across millions of apps. In addition to that, we are investigating methods to dynamically analyze pre-installed apps in virtualized environments, using Lumen to inspect closely their traffic.

References

[1] "OnePlus Secret Backdoor," https://www.theregister.co.uk/2017/11/14/oneplus_backdoor/, [Online; accessed 08-March-2018].

[2] N. Vallina-Rodriguez, J. Amann, C. Kreibich, N. Weaver, and V. Paxson, "A tangled mass: The android root certificate stores," in *Proc. ACM CoNEXT*, 2014.

[3] A. Razaghpanah, N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, P. Gill, M. Allman, and V. Paxson, "Haystack: In situ mobile traffic analysis in user space," *arXiv preprint arXiv:1510.01419*, 2015.

[4] K. Allix, T. F. Bissyandé, J. Klein, and Y. Le Traon, "Androzoo: Collecting millions of android apps for the research community," in *Proceedings of the 13th International Conference on Mining Software Repositories*. ACM, 2016, pp. 468–471.

[5] https://developer.android.com/guide/topics/permissions/overview.html, [Online; accessed 08-March-2018].

[6] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," *ACM SIGPLAN Notices*, 2014.

[7] "Tacyt, Eleven Paths, Telefónica," https://www.elevenpaths.com/technology/tacyt/index.html, [Online; accessed 08-March-2018].