AndroZoo: Collecting Millions of Android Apps for the Research Community

Kevin Allix, Tegawendé F. Bissyandé, Jacques Klein and Yves Le Traon
SnT, University of Luxembourg
4 rue Alphonse Weicker
L-2721 Luxembourg, Luxembourg
{kevin.allix, tegawende.bissyande, jacques.klein, yves.letraon}@uni.lu

ABSTRACT

We present a growing collection of Android Applications collected from several sources, including the *official* Google Play app market. Our dataset, AndroZoo, currently contains more than three million apps, each of which has been analysed by tens of different AntiVirus products to know which applications are detected as Malware. We provide this dataset to contribute to ongoing research efforts, as well as to enable new potential research topics on Android Apps. By releasing our dataset to the research community, we also aim at encouraging our fellow researchers to engage in **reproducible experiments**.

Keywords

Android Applications, APK, Software Repository

1. INTRODUCTION

Mobile app development has witnessed an unprecedented growth in recent years due to the increase in affordability and adoption of smart powerful handheld devices. In particular, the Android ecosystem, with its open Operating System and the available Software Development Kit, have empowered developers to produce millions of apps for diverse user tasks, ranging from mail and games to payment and health activities.

Unlike the few established traditional desktop applications which have been thoroughly studied by the research community, Android apps are legion, each having a large share of user base. Analysing these apps at a large scale is however challenging since market maintainers implement several restrictions in collecting apps. In this context, researchers proceed in a best effort way to reuse small datasets (which are generally obsolete), or collect a limited number of samples (which may not be representative), leading to studies which may be biased and experiments which are often not reproducible.

To address the problem of Android dataset collection, we have invested in a long-term effort to crawl apps for the research community. After several months of crawling, we have already stored over three million of apps. With Androzoo, we aim to provide the software engineering research community with an unrestricted, scalable and up-to-date access to Android apps. To that end, we have developed specialised crawlers for several market places to automatically browse their content, find Android applications that could be retrieved for free, and download them into our repository.

To the best of our knowledge, the total number of apps that we have collected constitutes the largest dataset of Android apps ever used in published Android research studies.

Often, it is impossible to know beforehand how many apps are available on a given market. Therefore, some of the markets for which we wrote dedicated crawlers proved to be much smaller than initially expected.

The crawlers we wrote follow two main objectives: a) Collect as many apps as possible, and b) Ensure the lowest possible impact on the market infrastructure. These two objectives increased the cost of writing such crawlers since for every market a manual analysis of the website has been performed in order to detect and filter out pages with different URL but with similar contents—for example lists that can be sorted according to different criteria. Similarly, a unique identifier for every APK on one market had to be found, so that deduplication can happen *before* downloading apps.

While reducing the load we incur to markets' web servers may not seem strictly necessary to the objective of collecting apps, it vastly reduces the likelihood of being banned by market owners and hence, helps building and maintaining in the long term a large and up-to-date dataset.

We present in this paper the architecture developed to collect the Androzoo dataset (cf. Section 2), then we discuss the challenges for setting up a working infrastructure (cf. Section 3). We also provide a few statistics on the dataset (cf. Section 4) before enumerating potential uses of Androzoo by the research community (cf. Section 5). Finally we discuss crawling limitations and provide concluding remarks.

2. CRAWLING ARCHITECTURE

We now provide details on the applications sources as well as on the multiple software components that were developed to build the collection and analysis infrastructure.

2.1 App Sources

2.1.1 Main Markets

MSR'16, May 14-15 2016, Austin, TX, USA

ISBN 978-1-4503-4186-8/16/05…$15.00

DOI: http://dx.doi.org/10.1145/2901739.2903508

Google Play. The official market of Android[1] is a website that allows users to browse its content through a web browser. Apps cannot however be downloaded through a web browser. Instead, Google provides an Android app[2] that uses a proprietary protocol to communicate with Google Play servers. No app, however, can be downloaded from Google Play without a valid Google account – not even free Apps. Both issues thus outlined were overcome using open-source implementations of the proprietary protocol and by creating free Google accounts. The remaining constraint was *time*, as Google also enforces a strict account-level rate-limit: a given account is not allowed to download more than a certain number of apps in a given time frame.

Anzhi. The anzhi market[3]—the largest alternative market of our dataset—is operated from China and targets the Chinese Android user base. It stores and distributes apps that are written in the Chinese languages, and provides a less-strict screening policy than e.g., Google Play.

AppChina. AppChina[4], another Chinese market, used to enforce drastic scraping protections such as a 1Mb/s bandwidth limitation and a several-hour ban if using simultaneously more than one connection to the service.

2.1.2 Other Android Markets

The **1mobile**[5] market proposes free Android apps for direct downloads: users can browse and retrieve thousands of apps. Other crawled markets are **AnGeeks** [6], and **Slideme**[7], which is operated from the United States of America.

FreewareLovers[8] is run by a German company, and provides freeware for every major mobile platform, including Android. **ProAndroid**[9], operated from Russia, is amongst the smallest markets that we crawled. It distributes free Apps only.

We also crawled **HiApk**[10] and **F-Droid**[11], a repository of Free and open-source software on the Android platform that provides a number of apps that users can download and install on their devices. Many of the applications found on F-Droid are modified versions of apps that are released to other markets by their developers. The modifications brought by F-Droid are usually linked with advertisement and/or tracking library removal.

2.1.3 Other sources

In addition to market places, we also looked into other distribution channels to collect applications that are shared by bundles.

Torrents. We have collected a small set of apps which were made available through BitTorrent. We note that such applications are usually distributed without their authors' consent, and sometimes include Apps that users should normally pay for. Nevertheless, when considering the number of leeches, we were able to notice that such collections of Android applications seemed to attract a significant number of user downloads, increasing the interest for investigating apps distributed in such channels.

Genome[12]. Zhou et al.[6] have collected Android malware samples and gave the research community access to the dataset they compiled. This dataset is divided in families, each containing malware that are closely related to each other.

2.2 Typical Crawlers

For most app sources, we developed a dedicated web crawler using the scrapy[13] framework. Every candidate app which is available for free runs through a processing pipeline that:

1. Ensures this app has not already been downloaded;
2. Downloads the file;
3. Computes its SHA256 checksum;
4. Archives the file.

To check that an app has not been already downloaded, we first identify a unique identifier for APKs in the market associated to the crawler, and store in a CouchDB[14] base an entry *market_name–App_identifier*. As a consequence, and because it is impossible to determine that two files from two markets are the same unless both are downloaded and compared, the deduplication is local to one market, meaning that one file from one market is downloaded exactly once, regardless of whether or not is has already been downloaded from another market.

2.3 Google Play Crawler

Google Play has several features that make automatically crawling it harder than other markets. As a result, a more elaborated crawler is required for this market. Amongst those features are the need for authentication with a valid Google Account currently associated with an Android device, the impossibility to obtain a list of all available applications and the necessity to use an undocumented protocol for communicating with Google Play servers. Google further enforces limits on the number of apps that can be downloaded per Google account in a given period from one IP address.

To overcome those limits, we wrote a software dedicated to finding and downloading apps from Google Play. This software is built with two components: a central dispatcher, and a download agent.

We have used agents on up to seven machines located in Luxembourg, France and Canada. On three of these machines, we ran two instances of the agent, one using exclusively IPv4 connectivity and the other using IPv6. Because IPv4 and IPv6 addresses are not linked in any way, this allows to hide the fact that those two agents run on the same machine, hence enabling us to increase the number of applications downloaded from one computer without increasing the risk of being blacklisted.

Our Google Play Crawler infrastructure managed to collect up to 296 448 new APKs in just one civil week, which demonstrates the ability of our software to easily cope with the volume of free applications published through the official market. Thus, after several weeks catching up with old applications, it appeared that two agents are sufficient to keep up with the flow of newly released apps.

2.4 Collection Manager

The collection manager is a web service responsible for all bookkeeping activities. It receives all the APKs that

[1] http://play.google.com (previously known as Google Market)
[2] Also named Google Play
[3] http://www.anzhi.com
[4] http://www.appchina.com
[5] http://market.1mobile.com
[6] http://www.angeeks.com
[7] http://slideme.org
[8] http://www.freewarelovers.com
[9] http://proandroid.net
[10] http://www.hiapk.com
[11] http://f-droid.org
[12] http://www.malgenomeproject.org
[13] http://scrapy.org
[14] https://couchdb.apache.org

were downloaded by crawlers, and stores them on the file system, handling safely the potential conflicts inherent to every parallel software.

It enables apps to be downloaded by authenticated users, and provides a web page detailing statistics on the whole dataset and on the recently added APKs.

This software component is written in Python using the Flask[15] framework. A PostgreSQL database accommodates data storage and querying needs, and embeds parts of the application's logic in PL/pgSQL functions.

3. DATA COLLECTION CHALLENGES

We ran into several difficulties while trying to maintain our infrastructure running. For example, two different markets were unreachable for a period of time longer than any expected maintenance-induced downtime: 1) for a full month, the 1mobile market was unavailable and then came back to normal; 2) The market apk_bang however completely disappeared just a few days after we started crawling it, never to come back online again.

Other more general issues are the following:

HTML Stability. During the time we collected applications, our crawlers had to be adapted about twenty times. Indeed, very often each market made changes to the structure of the HTML pages it generates. Most of the times, those changes implied that the XPath expressions used to scrap useful information from web pages had to be fully rewritten, which requires a new manual analysis of the web pages.

Monitoring Crawlers. Detecting that an HTML stability issue is happening may not always be straightforward. For smaller markets, it is not unusual to detect no new application during several days. This can have two possible explanations: Either no new application was offered by one given market—in which case our crawler is working as expected—or it could be that our crawler failed to detect and/or collect the new applications—which could mean an HTML stability issue happened.

Protocol Change. One market moved from a standard website where applications could be downloaded from a web browser to a model where applications could only be obtained through a dedicated, market-specific application. While we probably could have reverse-engineered the undocumented protocol used by that market application, we considered that it was not worth the effort and instead simply stopped collecting apps from this market.

Information Loss. Very few application sources allow users to download previous versions of a given app. Instead most markets only allow the latest version to be downloaded. Coupled to the fact that it is not unusual for apps to be updated several times a week, it is impossible to guarantee that all versions of all apps have been added to our collection. In addition, if a given version could be downloaded in time before it is replaced by a new version in the market, it will never be available again for collection.

4. ANDROZOO

Our dataset currently contains more than three million unique Android apps, adding up to more than 20 TB. The distribution of apps according to their source is shown in Table 1. The diversity of application sizes in our dataset is shown in Figure 1.

Table 1: Current state of the AndroZoo APK repository

Marketplace	# of Android apps	Percentage
Google Play	1 899 883	59.70%
Anzhi	605 646	19.03%
AppChina	577 662	18.15%
1mobile	57 525	1.81%
AnGeeks	55 804	1.75%
Slideme	52 145	1.64%
torrents	5 294	0.17%
freewarelovers	4 145	0.13%
proandroid	3 683	0.12%
HiApk	2 491	0.08%
fdroid	2 023	0.06%
genome	1 247	0.04%
apk_bang	363	0.01%
Total	**3 182 590 Unique apps**	

This dataset has been, and still is being built over time. As a consequence, the number of applications in the dataset is still growing.

4.1 Download API

The HTTP API that we provide allows to download full, unaltered APKs. Each APK is actually a zip file that contains: a dex file, which is the bytecode of the application, at least one cryptographic certificate that signed this app, various assets (images, audio files, libraries, etc), and a `Manifest` file.

On the AndroZoo website we provide a regularly updated list[16] of available APKs (identified by their SHA256 hashes), along with metadata on each app compilation date (dex_date), the number of antivrus engines which flag it as malicious (vt_detection), the size of the APK, the size of the dex code, the main package name, the version, and the market where the app was downloaded from. With this information, researchers can select the subset of apps which they would like to retrieve from AndroZoo using a dedicated API which allows to specify the SHA256 hash of an APK that is requested.

4.2 Example Statistics

We started crawling in late 2011, and have continued crawling since. Over time, we added more and more app sources, and optimised our crawlers' efficiency.

Figure 1 shows the long-tail distribution of the APK sizes in the AndroZoo dataset.

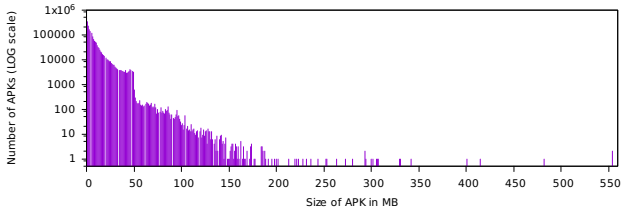

Figure 1: Distribution of APKs size

We have sent all apps in AndroZoo to VirusTotal, a web portal that hosts over 60 products from renown antivirus vendors, including McAfee, Symantec or Avast. Figure 2 shows the percentage of applications flagged by at least 1

[15] http://flask.pocoo.org/

[16] https://androzoo.uni.lu/lists

antivirus product for each of the 4 dataset sources presented. However, to stress out that results from antivirus must be considered carefully, we show in Figure 3 for each data source the percentage of applications flagged by at least 10 antivirus products. Only 1% of Google Play apps now remain in the category of malware.

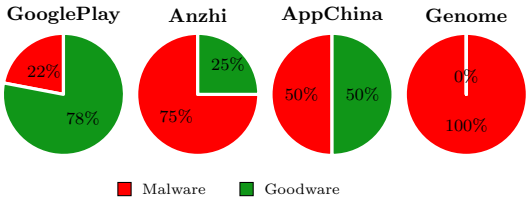

Figure 2: Share of Malware in Datasets: Applications are flagged by at least 1 antivirus product

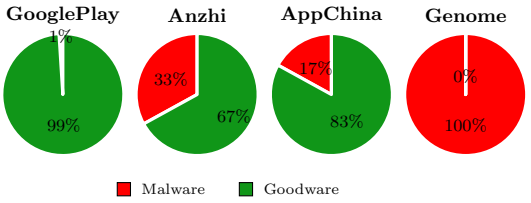

Figure 3: Share of Malware in Datasets: Applications are flagged by at least 10 antivirus products

4.3 Access Conditions

We make our dataset available to the research community. Given the lack of a clear, universal copyright exemption for Research, we **request** that researchers willing to access this dataset: 1) Evaluate the legal situation of downloading and working on copyrighted applications with regards to their situation (local laws, host institution policy, etc); 2) Do not, in general, redistribute the data; 3) Do not, in particular, make a commercial usage of this data; 4) Act responsibly with this data, notably with regards to the maliciousness of many apps; 5) Get a faculty, or someone in a permanent position, to agree and commit to those conditions.

We **politely ask** that the origin of the dataset be acknowledged, and we **hope** that researchers will make available the lists of apps used in their publications to make their experiments reproducible.

5. LEVERAGING ANDROZOO

This dataset has already been used to conduct research in the field of Machine Learning-based Malware Detection. In particular, the scale of this dataset allowed to demonstrate methodological issues when evaluating the performance of ML-based Malware Detector [1], to emphasise the importance of Time in malware detection experiments [2] and to draw a landscape of the Android Malware environment [3]. It also allowed to highlight the evolution of Android Apps over time [4] by finding *Antipatterns*, and to detect Privacy Leaks [5].

Potential usages could be found in the fields of Code Recommendation, large scale studies on API usage and adoption, coding patterns, repackaging detection, Library detection and popularity analysis, Obfuscation techniques, Malware analysis, application similarity, etc.

Our dataset, thanks to its long-term coverage, is particularly well suited to enable evolution studies. Indeed, in this dataset are more than 5 000 apps for which we have 15 or more versions.

6. LIMITATIONS

The number of apps in any given time frame may be more linked to the our collection process than to the overall appearance of apps during this time frame. Indeed, our crawlers are managed as a low-priority research project rather than as a mission-critical production system. Collecting was regularly interrupted for days, weeks, or even a few months, for issues such as lack of storage space or more generally, limited workforce to invest.

We have found that several market owners took various steps in order to prevent their market from being automatically mined. Thus, for such markets, we cannot guarantee that we have retrieved their whole content.

For obvious reasons, we only collect apps that can be downloaded for free.

7. CONCLUSIONS

We have presented the AndroZoo dataset of millions of Android apps collected from various data sources. We make this dataset readily available to the community to contribute to more generalisable, reliable and reproducible studies based on a large-scale, representative, and up-to-date samples. AndroZoo stats are available at: https://androzoo.uni.lu

8. REFERENCES

[1] K. Allix, T. F. Bissyandé, Q. Jerome, J. Klein, R. State, and Y. Le Traon. Empirical assessment of machine learning-based malware detectors for android: Measuring the gap between in-the-lab and in-the-wild validation scenarios. *Empirical Software Engineering*, pages 1–29, 2014.

[2] K. Allix, T. F. Bissyandé, J. Klein, and Y. Le Traon. Are your training datasets yet relevant? an investigation into the importance of timeline in machine learning-based malware detection. In *Engineering Secure Software and Systems*, volume 8978 of *LNCS*, pages 51–67. Springer International Publishing, 2015.

[3] K. Allix, Q. Jérome, T. F. Bissyandé, J. Klein, R. State, and Y. Le Traon. A forensic analysis of android malware: How is malware written and how it could be detected? In *Computer Software and Applications Conference (COMPSAC)*, 2014.

[4] G. Hecht, O. Benomar, R. Rouvoy, N. Moha, and L. Duchien. Tracking the software quality of android applications along their evolution. In *Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on*, pages 236–247, Nov 2015.

[5] L. Li, A. Bartel, T. F. Bissyandé, J. Klein, Y. Le Traon, S. Arzt, S. Rasthofer, E. Bodden, D. Octeau, and P. McDaniel. Iccta: Detecting inter-component privacy leaks in android apps. In *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on*, volume 1, pages 280–291, May 2015.

[6] Y. Zhou and X. Jiang. Dissecting android malware: Characterization and evolution. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, pages 95–109, Washington, DC, USA, 2012. IEEE.