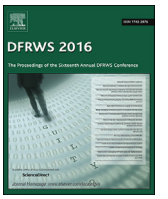
Digital Investigation 18 (2016) S33–S45

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin



DFRWS USA 2016 — Proceedings of the 16th Annual USA Digital Forensics Research Conference

Fingerprinting Android packaging: Generating DNAs for malware detection

ElMouatez Billah Karbab*, Mourad Debbabi, Djedjiga Mouheb

Computer Security Laboratory, Concordia University & NCFTA-Canada, Montreal, Quebec, Canada

Keywords:
Fingerprinting
Malware
Mobile
Android
Fuzzy hashing
Detection
Family attribution

A B S T R A C T

Android's market experienced exponential popularity during the last few years. This blazing growth has, unfortunately, opened the door to thousands of malicious applications targeting Android devices everyday. Moreover, with the increasing sophistication of today's malware, the use of traditional hashing techniques for Android malware fingerprinting becomes defenseless against polymorphic malicious applications. Inspired by fuzzy hashing techniques, we propose, in this paper, a novel and comprehensive fingerprinting approach for Android packaging *APK*. The proposed fingerprint captures, not only the binary features of the *APK* file, but also the underlying structure of the app. Furthermore, we leverage this fingerprinting technique to build ROAR, an automatic system for Android malware detection and family attribution. Our experiments show that the proposed fingerprint and the ROAR system achieve a precision of 95%.

Introduction

In recent years, we have witnessed a phenomenal popularity and growth of Android devices. It is estimated that 2 billion devices are currently powered by Android OS (Ericsson, 2013). This trend is expected to continue to reach more than 5.6 billion devices by 2019 (Ericsson, 2013). Due to their proliferation and ubiquitousness, Android devices have become a tempting target for cyber criminals. According to a *Cisco* report (Cisco, 2014), mobile malware mostly targets Android devices. In this setting, the need to develop effective and accurate forensics methods, techniques and tools for the detection and analysis of Android malware becomes a desideratum.

To address the malware variation flood, multiple defence mechanisms have been proposed by the *anti-mobile-malware* industry, with *signature-based detection* being the most adopted technique. The latter uses malware digest or *signature* to match against mobile applications in order to detect any malicious code. Traditional cryptographic hashing algorithms such as *SHA1* and *MD5* have been widely adopted for generating malware signatures. Cryptographic hashing methods have the advantage of being simple and fast. They are, however, highly sensitive to even small changes, which makes these methods defenseless against malware variations. Moreover, despite its effectiveness, signature-based detection could be easily defeated by new malicious applications with only tiny modifications as is the case with *polymorphic* attacks.

To overcome the drawbacks of cryptographic hashing, a new technique, namely *fuzzy hashing*, has emerged in the literature. The concept of fuzzy hashing was first introduced as *ssdeep* (Kornblum, 2006) in *rsync checksum* (Tridgell and Mackerras). The main advantage of this technique over cryptographic hashing lies in its tolerance to changes. Thanks to this important property, fuzzy hashing has been widely leveraged to detect Web-based document duplication (Figuerola et al., 2011). In cyber security, fuzzy hashing has been mainly embraced in malware fingerprinting. For instance, Virus Total has been using the *ssdeep* fuzzy hashing for malware fingerprinting since 2012. There

* Corresponding author.
E-mail address: e_karbab@encs.concordia.ca (E.B. Karbab).

http://dx.doi.org/10.1016/j.diin.2016.04.013

are several other attempts, such as *mvHash-B* (Breitinger et al., 2013), *dcfldd* (DCFL, 2016), *sdhash* (sdhash 2016; Roussev), and *mrshv2* (Breitinger and Baier, 2013), to apply fuzzy hashing in multiple applications. In the context of malware detection, it consists of two steps: i) malware binary digest computation, ii) matching the digest against other malware samples.

Despite its effectiveness compared to cryptographic hashing, fuzzy hashing technique suffers from some limitations. First, it ignores the underneath structure and semantics of the malicious package. Second, fuzzy hashing suffers from its single fingerprint bounded with a maximum size (e.g., *ssdeep* (Roussev, 2010)), thus preventing packages with different sizes and features to be effectively compared. Other drawbacks of fuzzy hashing related to specific algorithms are presented in (Li et al., 2015) concerning *mvHash-B* fuzzy hash algorithm (Breitinger et al., 2013). Furthermore, the compressed nature of Android *APK* package makes the repacking of malicious apps an easy task. Moreover, the increasing number of Android app stores and the lack of security verification in some stores increase the chance of attackers to deploy malicious applications in multiple stores. Another issue is related to *native libraries* (Wang and Shieh, 2015), specifically at the level of *Java objects*, executed on top of *Dalvik* machine. It has been shown that these libraries are exploited by a significant number of Android malware, such as the well-known sophisticated *DroidKungFu* malware and its many variations (Zhou and Jiang, 2012). Not analyzing the native library would make the distinction between its variations a very challenging task.

Our objectives are to: i) Develop a more accurate, yet broad, fuzzy fingerprinting technique for Android OS malware. The proposed fingerprint relies on a customized fuzzy hashing technique that addresses the previous limitations. ii) Design and implement a framework for Android malware detection and family attribution on top of the developed fuzzy fingerprint. To this end, we propose APK-DNA, a fuzzy fingerprint that captures both the structure and the semantics of the *APK* file using most Android *APK* features. This fingerprint covers: i) the underneath Android app structure, including both *Dalvik machine byte-code* ii) the meta-data of the Android app. Our empirical results indicate that our fingerprinting approach is highly robust to app changes and accurate in terms of fingerprint computation and matching.

Moreover, we build ROAR, an automatic framework for Android malware detection, using the proposed APK-DNA. The goal is to generate fingerprints for known Android malicious apps, and then detect new malware variations using similarity computing. In ROAR framework, we propose two different approaches for malware detection: i) *family-fingerprinting*, and ii) *peer-matching*. In addition to malware detection, we aim to attribute the family lineage of the mobile malware. To this end, ROAR attributes a similarity score to each possible variation in the same malware family. We evaluate APK-DNA and ROAR using real malware samples from the Android Malware Genome Project (Android Malware Genome Project, 2015; Zhou and Jiang, 2012). We experiment with multiple Android malware families. Our evaluation demonstrates that ROAR is highly accurate compared to state of-the-art approaches.

This paper makes the following contributions:

- We propose a novel and rather comprehensive fingerprinting technique for Android application packages (APK) based on fuzzy hashing. The proposed fingerprint considers not only the binary format of *APK* but also its structure and semantics.
- We design and implement ROAR, a framework that leverages the proposed fingerprinting technique for Android malware detection, following two different approaches, namely *peer-matching* and *family fingerprint*. In addition, ROAR is able to detect malware variations and attribute the family of the detected malware.
- We evaluate ROAR on 928 Android malware samples from (Android Malware Genome Project, 2015; Zhou and Jiang, 2012) dataset. The evaluation results demonstrate the high accuracy of ROAR in terms of both malware detection and family attribution.

The remainder of this paper is organized as follows: Section Approach overview presents an overview of the proposed approach. Section APK-DNA fingerprint is dedicated to the APK-DNA fingerprinting. Section ROAR framework presents the ROAR framework. Section Experimental results details our experimental results. Section Limitations and future work discusses the limitations of the proposed approach together with some ideas on future research. The related work is reported in Section Related work. Section Conclusion contains some concluding remarks.

Approach Overview

Current fuzzy fingerprints such as *ssdeep* are computed against the app binary as a whole, which makes them ineffective for detecting malicious app variations. This problem gets even worst in case of Android OS because of the structure of apps packaging, which contains not only the actual compiled code but also other files such as media ones. To overcome this limitation, we propose an effective and broad fuzzy fingerprint that captures, not only binary features, but also the underneath structure and semantics of the *APK* package.

Accordingly, our approach for computing Android app fingerprints relies on decomposing the actual *APK* file into different content categories. For each category, we compute a customized fuzzy hash (sub-fingerprint). Note that for some categories, for instance *Dex* file, the application of the customized fuzzy hashing on the whole category content does not capture the structure of the underlying category. In this case, we apply fuzzy hashing against a selected *N-grams* of the category content. In our context, we use *byte n-grams* on binary files and *instruction n-grams* on assembly files. Furthermore, a best practice for malware fingerprinting is to increase the entropy of the app package content (Masud et al., 2007). To this end, we compress each category content before computing the