You Can't Touch This

Consumer-Centric Android Application Repackaging Detection

Iakovos Gurulian
Supervisor: Dr Konstantinos Markantonakis

ABSTRACT

Repackaged applications are modified versions of original applications, that can potentially target large audiences based on the original application's popularity. In this paper, we propose an approach for detecting repackaged applications. Our approach takes advantage of the attacker's reluctance to significantly alter the elements that characterise an application without notably impacting the application's distribution. These elements include the application's name and icon. The detection is initiated from the client side, prior to an application's installation, making it application store agnostic. Our experimental results show that detection based on our algorithm is effective and efficient.

PROBLEM

Application repackaging is a widely used method for malware distribution, revenue stealing and piracy. Detecting repackaged applications can be very challenging.

SOLUTION

Basing the detection on elements of the application that an attacker is reluctant to significantly alter allowed us to detect repackaged applications at a very high rate.

Application Repackaging

- Repackaged applications are applications that impersonate a genuine application, by slight modifications/variations to the genuine application's artwork and/or changes to its source code in a way that the repackaged application looks and/or feels like the genuine application. The main objective of a repackaged application is to mimic a genuine application so it can target novice users that gravitate towards the popularity/functionality of the genuine application. Applications that infringe the potential intellectual property of the original application and present themselves as unique/different applications are excluded from this definition.
- Application repackaging usually involves decompilation of the original application, modification, recompilation and signing with the attacker's key.
- 86% of all malware distribution relies on repackaged applications.
- Part of the OWASP's "*Top Ten Mobile Risks"* for 2014.

Threats

- **To the developer**
  1. Unauthorised redistribution
  2. Advertisement revenue stealing
  3. Cracking (Piracy)
- **To the user**
  1. Trojan Horse
  2. Denial of application upgrade

Project Goals

- Application download source independent, proactive detection
- Fast detection
- Do not rely on application stores/developers to take actions
- Initiate process from the client side
- Respect the user's bandwidth

Proposed Solution

- **Attacker's weak point:** *If data that define and distinguish the application (its name and its icon) get significantly altered during the repackaging process, it is not more likely to get installed on a victim's device than any random application.*
- Maintain a database with verified legitimate applications.
- Prior to the installation process a device extracts and sends relevant application information to the server.
- In case the application does not exist on the server, image and string similarity algorithms are used to retrieve perceptually similar applications.
- The server makes a decision and returns it to the device

Detection Process

- Five elements are extracted from the tested application prior to installation:
  1. Hash of the (signed) application
  2. Hash of the unsigned application
  3. Hash of the developer's signature
  4. Application name
  5. Application icon
- Data is sent to a trusted server
- The server processes the data and returns a decision

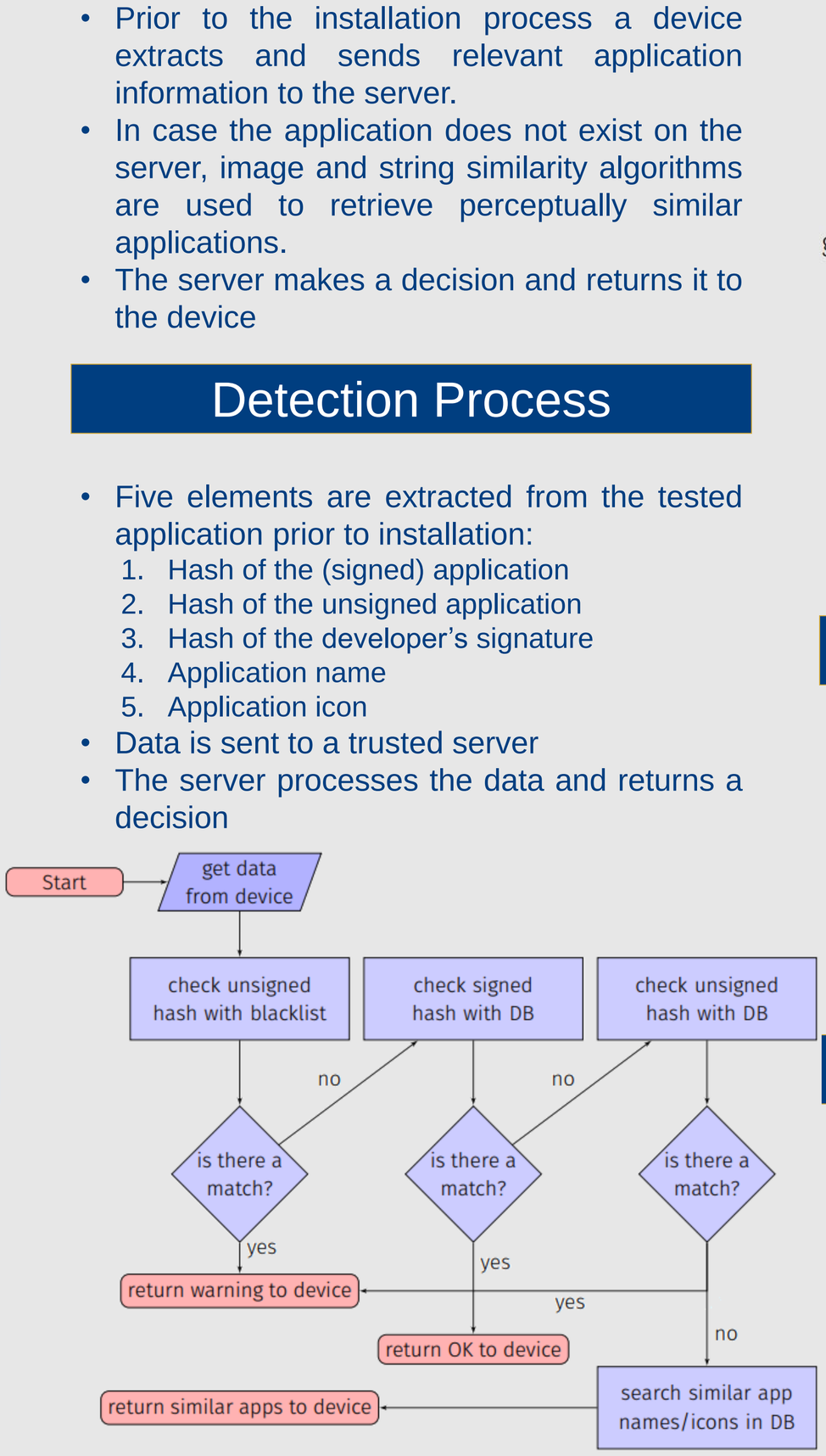

Figure 1: The detection process

Results

- 91.5% detectability rate
- Low false positive/false negative rates
- Detectability of applications that only clone the original application's name and icon (no source-code) – ***Example:*** *Figure 2*
- We noticed that attackers very rarely alter the names/icons of repackaged applications significantly

Figure 2: Real world example
*Upon checking an application with name **"googl app stoy"**, the server detects that an application with similar name/icon pair exists in the trusted database and returns it.*

Compared to Previous Works

- As accurate
- Much faster
- Application download source independent
- Detects applications that only clone the name and the icon
- Cheaper (money and resource wise)

Algorithms

- **String similarity:** Jaro-Winkler
- **Image similarity:** Haar wavelet
- **Similarity between applications** a **and** b:

$$Sim_{a,b} = 10\frac{x \times 10^{x} + y \times 10^{y}}{2}$$

Where x is the name similarity and y is the icon similarity

2015, Smart Card Centre – Information Security Group
Iakovos.Gurulian.2014@live.rhul.ac.uk | www.scc.rhul.ac.uk