COMPUTERS & SECURITY 65 (2017) 90–107

Available online at www.sciencedirect.com

ScienceDirect

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

Computers & Security

Android resource usage risk assessment using hidden Markov model and online learning

Bahman Rashidi [a,*], Carol Fung [a], Elisa Bertino [b]

[a] Department of Computer Science, Virginia Commonwealth University, Richmond, VA, USA
[b] Department of Computer Science, Purdue University, West Lafayette, IN, USA

ARTICLE INFO

Article history:
Received 22 June 2016
Received in revised form 14 September 2016
Accepted 10 November 2016
Available online 17 November 2016

Keywords:
Smartphone
Permission
App behavior
Risk computation
Privacy

ABSTRACT

With Android devices users are allowed to install third-party applications from various open markets. This raises security and privacy concerns since the third-party applications may be malicious. Unfortunately, the increasing sophistication and diversity of the malicious Android applications render the conventional defenses techniques ineffective, which results in a large number of malicious applications to remain undetected. In this paper we present XDroid, an Android application and resource risk assessment framework based on the Hidden Markov Model (HMM). In our approach, we first map the applications' behaviors into an observation set, and we attach timestamps to some observations in the set. We show that our novel use of temporal behavior tracking can significantly improve the malware detection accuracy, and that the HMM can generate security alerts when suspicious behaviors are detected. Furthermore, we introduce an online learning model to integrate the input from users and provide adaptive risk assessment. We evaluate our model through a set of experiments on the DREBIN benchmark malware dataset. Our evaluation results demonstrate that the proposed model can accurately assess the risk levels of malicious applications and provide adaptive risk assessment based on user input.

1. Introduction

The number of the global smartphone users is growing rapidly and is expected to double in the next years from 2.7 billion in 2016 to 6 billion in 2020 (Lunden, 2015). On the other hand, the number of mobile apps has been growing exponentially in the past few years. According to the report by Android Google Play Store, the number of apps in the store has reached 1.8 billion in 2015, surpassing its major competitor Apple App Store (Statista, 2015). As the number of smartphone apps increases, privacy and security have become a primary concern, especially when smartphones are used for sensitive tasks or business purposes. A malicious third-party app can not only steal private information, such as contact lists, text messages, and GPS locations from a user, but also causes financial loss for the user by making secretive premium-rate phone calls and text messages (Rothman, 2011; Yang et al., 2015).

In current Android architecture, apps have to request permissions in order to access phone resources such as GPS, contact lists, and files. Users decide whether to give out those permissions to apps. However, such an approach has been proven ineffective to protect users since users tend to rush through their decisions and grant all permissions to the apps they desire to install. Studies have shown that a significant percentage (>70%) of smartphone apps request additional permissions beyond their actual need (Gunasekera, 2012; What is the price of free). For example, a puzzle game app may request SMS and phone call permissions. Without sufficient information, it is difficult for users to make appropriate decisions. An

* Corresponding author.
E-mail addresses: rashidib@vcu.edu (B. Rashidi), cfung@vcu.edu (C. Fung), bertino@cs.purdue.edu (E. Bertino).
http://dx.doi.org/10.1016/j.cose.2016.11.006
0167-4048/

effective malicious app detection can provide additional information to users and prevent them from privacy breach (Lindorfer et al., 2014).

Depending on the technology used, malware detection techniques can be divided into static and dynamic methods, where the former focuses on static code analysis of apps' and the latter investigate apps' maliciousness at runtime (Enck et al., 2011; Faruki et al., 2015; Singh et al., 2016). One major drawback of static analysis is that it does not detect vulnerabilities introduced at runtime (Enck et al., 2010; Gosain and Sharma, 2015). Dynamic analysis identifies vulnerabilities at runtime and supports the analysis of applications without actual code. It also identifies vulnerabilities that might have been false negatives in static code analysis (Ball, 1999; Enck et al., 2011). In this paper, we analyze Android apps' behaviors as they run on the device, and propose XDroid, a dynamic analysis method based on Hidden Markov Models (HMM) (Khreich et al., 2012). A HMM engine can be used to model the runtime behaviors of an app, including malicious and normal ones. The use of HMM for malicious apps detection has already been proposed by Chen et al. (2014). However such approach suffers from low detection accuracy. One major reason is that their approach only considers the apps' intents as observations. In our approach, we consider other inputs such as API calls, time, ads libraries, and sensitive permission requests to build a comprehensive HMM. We discovered that the introduction of the time feature significantly improves the detection accuracy of the model. In our approach, we first log apps' behaviors through an instrumentation tool that we developed, by our research lab, called *DroidCat*. Then a filtering and parsing method is applied to synthesize and organize the captured behaviors. We train and test the HMM model using a dataset of known malicious apps and normal apps. Our experimental results demonstrate that our proposed model achieves high accuracy in detecting malicious apps.

The major contributions of the work reported in paper include: (1) An instrumentation tool that facilitates app behavior logging in order to generate high quality dataset for analysis. (2) A comprehensive time-aware Android app behavior analysis, which is based on the apps' intents and actions, as well as extra features that further improves detection accuracy. (3) A trained hidden Markov model which can decide whether an app is malicious or not based on its behavior. (4) A dynamic model which can be updated in real time to integrate users' preferences. To the best of our knowledge, this is the first time that a HMM online learning model is used on malicious app control in smartphone security.

The rest of the paper is organized as follows: Section 2 describes the system design; Section 3 describes our proposed model, and the approach we use for capturing apps' logs, training and testing the model, and the on-line parameter update strategy; we present our evaluation results and the impact of the risk computing parameters in Section 4; related work is discussed in Section 5; Section 6 discusses key aspects of our approach, and finally Section 7 concludes the paper.

2. System design

The ultimate goal of XDroid is to monitor the behavior of apps and generate alerts to users when suspicious app behaviors are detected. Fig. 1 shows the architectural design of XDroid. The system contains components on the server side and the mobile device side. Each XDroid device contains an *Interaction Portal* and an *Activity Logger*. The interaction portal provides an interface for users to interact with the device. The activity logger is used to monitor the activities of the apps. The server side components include *Risk Assessment*, *User Profiling*, and *Alert Customization*. In the rest of this section, we describe the key features of the server.

2.1. *Interaction portal*

The interaction portal is to facilitate the interaction between users and devices. Instead of sending requests to the Android system's legacy permission handler (e.g. Package Manager Service), the XDroid handles the permission requests through the process illustrated in Fig. 2. For example, when a user installs "Telegram" (a popular messaging application) and choose to monitor its behavior using XDroid, the requested resources are displayed along with their estimated risk levels (Fig. 3(a)). The user can check resources he/she want to monitor. If a resource is monitored and suspicious activities related to it are detected, the user is informed through a dialog box (Fig. 3(b)). The user can decide whether to block the resource access or allow it based on the estimated risk suggested by XDroid.

For each installed app, the user can use the pre-installed XDroid application to view a list of apps which are under monitoring. If the user clicks on an app in the list, a set of requested

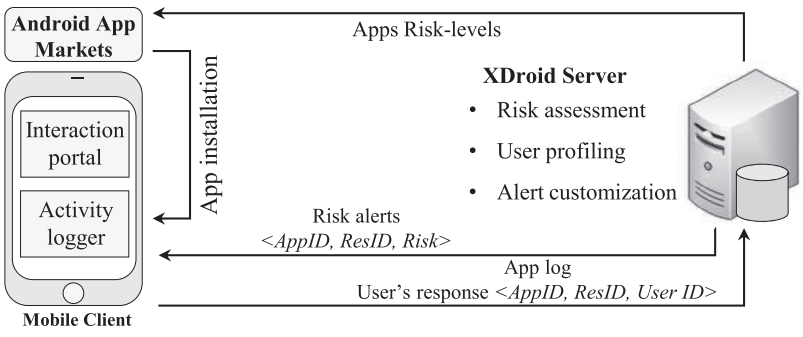

Fig. 1 – XDroid system overview.