Information and Software Technology 75 (2016) 17–25

Contents lists available at ScienceDirect

Information and Software Technology

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

Effective detection of android malware based on the usage of data flow APIs and machine learning

Songyang Wu, Pan Wang, Xun Li, Yong Zhang*

The Third Research Institute of Ministry of Public Security, Shanghai 201204, China

ARTICLE INFO

Article history:
Received 27 July 2015
Revised 19 February 2016
Accepted 14 March 2016
Available online 16 March 2016

Keywords:
Android security
Malware detection
Privacy leakage

ABSTRACT

Context. Android has been ranked as the top smartphone platform nowadays. Studies show that Android malware have increased dramatically and that personal privacy theft has become a major form of attack in recent years. These critical security circumstances have generated a strong interest in developing systems that automatically detect malicious behaviour in Android applications (apps). However, most methods of detecting sensitive data leakage have certain shortcomings, including computational expensiveness and false positives.

Objective. This study proposes an Android malware detecting system that provides highly accurate classification and efficient sensitive data transmission analysis.

Method. The study adopts a machine learning approach that leverages the use of dataflow application program interfaces (APIs) as classification features to detect Android malware. We conduct a thorough analysis to extract dataflow-related API-level features and improve the k-nearest neighbour classification model. The dataflow-related API list is further optimized through machine learning, which enables us to improve considerably the efficiency of sensitive data transmission analysis, whereas analytical accuracy is approximated to that of the experiment using a full dataflow-related API list.

Results. The proposed scheme is evaluated using 1160 benign and 1050 malicious samples. Results show that the system can achieve an accuracy rate of as high as 97.66% in detecting unknown Android malware. Our experiment of static dataflow analysis shows that more than 85% of sensitive data transmission paths can be determined using the refined API subset, whereas time of analysis decreases by nearly 40%.

Conclusion. The usage of dataflow-related APIs is a valid feature for identifying Android malware. The proposed scheme provides an efficient approach to detecting Android malware and investigating privacy violations in malicious apps.

1. Introduction

Today, smartphones are not only phones but also act as portable computers that provide diverse services such as short messages service (SMS), email, Internet browsing, social networking, maps, GPS, and mobile payment applications. As the most popular smartphone platform, Android is run on approximately 52.61% of smartphones worldwide as of October 2015.[1] Numerous android applications (apps) are available through various application sources (e.g., Google Play Store) . However, smartphones running Android are also targeted by malware [1]. While users enjoy the convenience of Android smartphones, hidden malware may steal stored sensitive privacy data such as contacts, SMS messages, and location information, for financial gain or other purposes. Recent studies [2–6] have examined different types of Android malware and determined that the most prevalent and serious is privacy violation, in which sensitive data are leaked to attackers. Furthermore, more dangerous attacks such as botnets, fraud phishing, or information theft, can utilize such stolen personal data. To reduce security threats to users, efficient analytical tools that can identify and defend against privacy leakage are required.

State-of-the-art approaches against privacy leakage in Android apps use data flow analysis to identity whether or not sensitive data leaves the device [7–11]. These schemes label data from

* Corresponding author. Tel.: +8602168571226.
E-mail address: zhangyonglab@yeah.net (Y. Zhang).

[1] http://www.netmarketshare.com

http://dx.doi.org/10.1016/j.infsof.2016.03.004
0950-5849/

sensitive data sources and track data propagated through multiple variables. When the labelled data leaves the system, the sensitive dataflow path is recorded including a source point, data propagation path, and sink point. However, reading and sending sensitive data are common behaviours in both benign and malicious apps. Many popular apps that provide interesting functions such as a cloud service, social networking, and mobile payment normally require sensitive data collection (e.g., location and contacts). This may results in numerous private data leakage alarms during malware detection. For example, we received more than 700 sensitive privacy leakage alarms during an experiment in which we conducted an analysis on a benign mobile payment app (com.unionpay.mobilepay.mpos.Activity.apk) by applying the open source taint analytical tool FlowDroid [8]. When a human analyst is drowned in such an extensive number of leakage alarms, identifying effectively those that are real malicious data transmission behaviours is difficult. Computational cost is another problem. Static taint analysis schemes always use a long list of source and sink APIs to scan decompiled code and detect data transmission paths between sources and sinks. Longer API list means increased computational overhead. In our experiments, Amandroid [9] and FlowDroid [8] (using default configure settings) often require hours to complete analysis in some real-world Android apps.

The purpose of this study is to determine the means of identifying Android malware and their malicious sensitive data transmissions more effectively than the current methods. We build a robust method for Android malware detection by applying machine learning in which dataflow-related API-level features are employed to distinguish malware from benign apps. The term 'dataflow-related API' has a similar meaning as the aforementioned 'source and sink' system calls. Our scheme relies on 'dataflow-related' API-level features because they can convey the major semantics of app behaviours on sensitive data. Our experiments also indicate that the proposed approach is effective.

The main contributions of this study are as follows:

- The proposed scheme combines machine learning and static dataflow analysis technologies which can accurately identify Android malware and efficiently discover sensitive data transmission paths. Approaches to extracting dataflow-related API-level features accurately through static program analysis are discussed in detail.
- We calculated a weight value for each API according to its use in malware. Based on the calculated malicious weights set, an improved Mahalanobis distance is proposed to compute the distance of neighbouring nodes in the k-nearest neighbour (KNN) algorithm. This notably promotes the accuracy of our classifier. An 'important API' (that scores a high malicious weight) list is further adopted to reduce dramatically the overhead in terms of time of static privacy leakage analysis.
- The proposed system is evaluated using more than 2200 real-world Android apps. The results show that our approach can achieve an accuracy that is as high as 97.66% in detecting unknown Android malware. In addition, the time overhead of static privacy leakage analysis is reduced by nearly 40%.

2. Related works

In recent years, many techniques and schemes have been proposed to resolve the growing problem of Android malware.

One much-studied direction focuses on detecting sensitive information transmission. Yang et al. [7] identified privacy leakage based on whether sensitive data transmission is user intended. Their research systematically studied means of distinguishing user-intended from unintended sensitive transmissions and thus provides a useful automated tool for identifying legitimate transmissions. Static taint analysis techniques [8,9] adopt graph reachability analysis and program slicing to identify possible privacy leakage paths. FlowDroid [8] is a static taint analysis system that is fully sensitive to Android context and objects. FlowDroid analyses configuration files and decompiles an Android app in order to construct an inter-procedural control-flow graph, which is used to find potential privacy leaks that are either caused by carelessness or malicious intention. Amandroid [9] focuses on inter-component static dataflow analysis, which constructs an inter-component dataflow graph (IDFG) and data dependence graph (DDG) for an Android app. Certain security problems, such as data leakage and data injection, can be reduced to query DDG and IDFG. However, building IDFG and DDG incurs extremely high overhead. In experiments conducted on a server with a 2.26 GHz quad-core Xeon CPU and 64 GB RAM, we adopted Amandroid to analyse a test set of Android apps (2–5 Mb in size, downloaded from popular markets). However, more than half of the test cases failed because of memory overflow or timeout. Dynamic taint analysis techniques [10,12] facilitate original app code and detect sensitive data propagation during app execution. TaintDroid [10] can simultaneously track multiple sources of sensitive data. In the approach of Zhang et al. [12], users can receive a notification when certain suspicious behaviour occurs. However, one inherent undesirable characteristic of such a dynamic detecting system is that code coverage cannot be guaranteed. For example, many apps often require users to register and login to activate full functionalities, which obviously hinders a complete dynamic analysis.

Another research direction of related studies is to detect possible malicious behaviour by applying machine learning. Our work falls into this category. Classification models based on system calls are common approaches for characterizing the behaviour of programs [13]. Crowdroid [14] collects system call traces of running apps on different Android devices and applies clustering algorithms to detect malware. Peng et al. [15] applied probabilistic learning methods to calculate risk scores according to the requested permissions of an Android app and identified a hierarchical mixture of naive Bayes as the best classifier of detecting tasks. DroidAPIMiner [16] analyses relevant features of malware behaviours captured at requested permissions, critical API calls, package-level information, and app parameters. This program evaluates several machine learning classifiers and achieves a 97.7% detection rate on malware samples using KNN. Drebin [17] adopts a hybrid approach to extract features from requested permissions and API calls as malware characteristics. Drebin uses support vector machines as classifiers and achieves a 94% detection rate on malware samples. Elish et al. [18] proposed a highly accurate classification approach. They proposed a scheme that statically extracts a property called user-trigger dependence as a classification feature. This property includes dataflow features related to the manner in which a user inputs trigger sensitive API invocations. Overall, previous approaches that used machine learning mainly focused on accurately detecting Android malware. Certain critical aspects, such as seeking sensitive data leakage and digital forensics of malicious behaviour, were not considered. Compared with the aforementioned existing schemes, our work verifies that the usage of dataflow-related APIs is a valid feature for identifying Android malware. Moreover, we used the results of a classification model to reduce significantly the computational overhead of statics taint analysis.

3. Design

We adopt a generic data mining approach to build our Android malware detection system. The system consists of two phases: training and identification. As illustrated in Fig. 1, the system in the training phase takes both benign and malicious Android apps