A System for Privacy Information Analysis and Safety Assessment of iOS Applications

Bin Li[1,2(✉)] and Zhijie Feng[1,2]

[1] Data Assurance and Communication Security Research Center, Beijing, China
libin@iie.ac.cn
[2] State Key Laboratory of Information Security, Institute of Information Engineering, CAS, Beijing, China

Abstract. As the intelligent mobile phone stores more and more users' privacy information, the privacy information security in mobile terminal has become more and more important to mobile users. So how to protect the sensitive information of mobile phone users has become the focus of research in recent years. This paper analyzed the security mechanism of iOS platform and security status and development of iOS Apps, and then proposed a scheme–iOS software sensitive information analysis and security assessment system, for detection of iOS analysis software of leaking sensitive information. The system is used to detect whether APPs in iOS platform is revealing user privacy data, and make evaluation of safety grade according to the severity of privacy disclosure of APPs. We give an evaluation algorithm about security level for evaluating the severity of APPs. This system is the overall solution of the iOS application software acquisition, privacy leak detection and security assessment.

Keywords: iOS · Privacy leak · Dynamic analysis · Security level

1 Introduction

With the popularization of mobile intelligent devices, and continues to increase of hardware performance, the intelligent mobile terminal are increasingly used to deal with all kinds of important data. While currently the most intelligent terminal operating system are focus on iOS and Android. With the popularity of iPhone mobile phone and iPhone "legalization of jail break" means, iPhone mobile phone security issues are also increasingly prominent. A recent research report said that 96 % of iOS apps have the ability to access sensitive information (contact information, calendar details, or location) from the device [1]. And many of them have privacy leaking or other security problems. In order to ensure safety of user information, iOS designed its own set of security mechanism:

Trust Boot. The system boots from the boot program, loading the firmware, and the firmware starts the system. The firmware via the RSA signature, only through the verification can get to the next step, and then to the firmware verification. In this way, the system establishes a trust chain at the root of bootstrap program.

J. Tian et al. (Eds.): SecureComm 2014, Part II, LNICST 153, pp. 392–398, 2015.
DOI: 10.1007/978-3-319-23802-9_31

Program Signature. The file format of iOS APPs is Mach-O format file [2]. This file format supports encryption and signature, and the directory structure is stored in memory through the SHA-1 hash. Both directory and software must be digitally signed.

SandBox. The application sandboxing has been defined by Apple as a set of fine-grained control that limits the application access to the file system, network and hardware [3]. Apple uses SandBox to isolate applications and limit the access of each process to the file system. Each application has its own memory space, and IOS applications can only read the file which is created for the program while other applications cannot access. Thus, all the application requests must be through the authority detection.

Address space layout randomization (ASLR) is one of the security protection technology and ASLR makes address predicting more difficult by randomizing the location of objects in memory. In iOS, the location of the binary, libraries, dynamic linker, stack, and heap memory addresses are all randomized [4].

Key chain and data protection. Password, certificate, keys are stored in Sqlite database, and data is encrypted in the database which has the strict access control.

While the mobile operating system need to protect information such as the mobile phone number, mail list, SMS, account passwords and other private information, so it's needed to be more careful in guarding privacy protection. However, Apple Corp does not pay much attention on the privacy protection. Although the Apple will ask users if they agree APPs to access their mail list, tracking the location information, and even the use of camera in iOS6 and future version [5], the user privacy security is likely to remain unresolved. Because you cannot determine whether these data acquired by APPs is stored encrypted or transmission. In the application software, the Apple Corp mainly relies on their own software review mechanism to guarantee the security of iOS software, and the application can be published to the App Store only by approval, so strict inspection prevents malicious programs to the system. However, millions of jail-break mobile phone users download APPs not only from App Store, but also from non-official app stores that almost never do any strict and meticulous examination of their applications. Even some companies steal user privacy information for their personal intention. Thus, privacy leak has become a universal phenomenon in mobile applications, a serious threat to the security of user data. Therefore, we need to design a set of detection system of sensitive information for iOS platform, automatic detect on the application of APPs in the market stores, determine the privacy problems of each APPs, in order to ensure the security of user information.

2 Privacy Information Analysis and Safety Evaluation System

In this paper we design and implement a system that makes a set of automatic detection to apps for sensitive/privacy information on the iOS platform. The system can provide awareness of potential user privacy leaks, such as contact

information, phone numbers, text message, user account, password, etc. and prompted user the potential risk of software. And then we store and analyze the results, define the security level according to a specific algorithm of software security ranking for each app. The system is mainly divided into three parts: app acquisition platform, app analysis platform and information management platform.

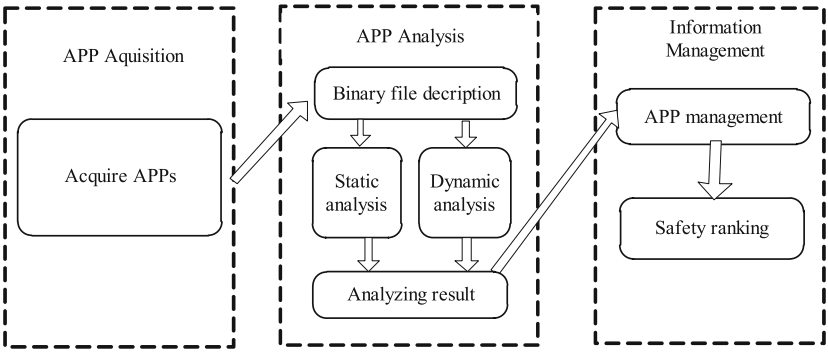

Fig. 1. System composition

As is shown in Fig. 1, app acquisition platform is responsible for collecting apps from the app store and store markets. The apps analysis platform is mainly for app decryption and analysis. Because all the apps that stored in app store have been certified with signature authentication and their code section are encrypted. So we need binary decryption of apps before analysis. App analysis includes dynamic and static analysis. The system checks all the APIs that may result in user privacy leak, and transfer analyzing results to the information management platform. Information management platform store security information of the apps, and calculate safety score by privacy scoring algorithm.

2.1 Binary File Decryption

Since the code segment of the apps have been encrypted by Apple, we cannot view the assembly code and call stack through the disassembly tool directly. We need to make decryption operations to apps. The executable file of iOS apps are Mach-O file, which copyrighted content with DRM [6], so the key is to decrypt the encrypted part of Mach-O file. When the program runs in the mobile phone, program is in decrypted state in memory. We must determine the position where the encryption code section locates in memory. We can use otool to see the FAT of the binary file. We need three parameters: cryptid, cryptoffset, cryptsize. cryptid is encryption state, 0 represents no encryption, 1 represents encrypted; cryptoffset represents offset of encrypting section; cryptsize means size of encrypting section in bytes. After located encrypted code section, we

export it through gdb tools, and replace the code section. Finally, we finish the decryption by recomposing the binary.

2.2 Static Analysis

After we decrypted the binary file of the app, we step into static analysis. Static analysis subsystem first disassembled the decrypted app, to get the APIs and selector references of the app, and by comparing the APIs in the app and in "local privacy API repository" (we need to create it in advance), the subsystem tells that whether the app binary contains privacy APIs. And we can make a conclusion that one app is safety if it does not contains any privacy APIs. The flow of static analysis is as the following chart.

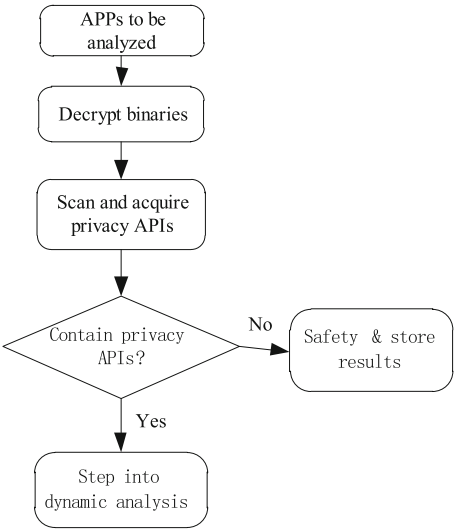

Fig. 2. Static analysis

2.3 Dynamic Analysis

In the dynamic analysis, we mainly use the behavior analyzing technique. We analyze the apps' behavior in the running process including when and how to evoke these privacy APIs. Dynamic analysis system detects APPs running on iOS devices and tracks the program running behavior real-time. It implements the real-time tracking of privacy APIs using MobileSubstrate/CydiaSubstrate framework. Cydia Substrate (formerly called MobileSubstrate) is the de facto framework that allows 3rd-party developers to provide run-time patches to system functions [7]. We can write hook function within the framework, and we

add tracking detection in the hook function by implementing MSHookMessageEx/MSHookFunction. MSHookMessage() will replace the implementation of the Objective-C message -[class selector] by replacement, and return the original implementation, while MSHookFunction()is like MSHookMessage() but is for C/C++ functions. When these hook functions are added into Dynamic libraries of iOS devices, the framework will invoke the Dynamic libraries automatically to implement the tracking procedure. The hook function is implemented by MobileLoader in MobileSubstrate framework. MobileLoader loads 3rd-party patching code into the running application [7]. An example will show the following substitution:

```
static CFArrayRef (*original_ABAddressBookCopyArrayOfAllPeople)(ABAddressBookRef addressBook);
CFArrayRef replaced_ABAddressBookCopyArrayOfAllPeople (ABAddressBookRef addressBook)
{
    printf("function:  ABAddressBookCopyArrayOfAllPeople called!!\n");
    original_ABAddressBookCopyArrayOfAllPeople(addressBook);

}
MSHookFunction(ABAddressBookCopyArrayOfAllPeople,
               replaced_ABAddressBookCopyArrayOfAllPeople,
               &original_ABAddressBookCopyArrayOfAllPeople);
```

Fig. 3. API subtitution

In above example we track API of AddressBook achieving. When the app has invoked this API while running in the iphone, the API will be recorded. Thus, we need to build a repository for storing all possible privacy APIs and networking interfaces, and tag them in hook function; the MobileSubstrate framework to replace the memory code, the privacy information of the API will be recorded into a file for further analysis.

2.4 Security Level Scoring Algorithm

We use the Security level scoring algorithm for computing the security level of target apps. According to importance of privacy information in mobile phone we define the corresponding values for each privacy API. When the application leaks privacy data, the corresponding values will be recorded. Finally, we add up all the values and get a total score. We judge the security level according to the total score. We define the Security level scoring algorithm:

$$\sum_{i=0}^{n=item.length} score = factor1 * weight1 + factor2 * weight2 + ... + factorn * weightn + ScoreInternet \quad (1)$$

ScoreInternet means the app invokes the Internet interface to exchange data. At this point, we need to analyze the content of data packets. We think that one operation is risk when the data packets contains call records, contact records, phone number and other sensitive/privacy information. Calculating scores of this part as follows:

$$ScoreInternet = (10 + factori + factorj) * Maxi, j(weight) \tag{2}$$

If the app does not send any data through network, the ScoreInternet = 0. The following table lists the contents of sensitive information and the corresponding score:

Sensitive information	Value	Weight	Remarks
phoneRecord	15	3	Call records
Contactbook	15	4	Contact book
smsRecord	20	4	SMS
Location	5	2	Location information
IMEI	5	2	Mobile phone device code
Calendar	5	1	Calendar
IMSI	5	2	International mobile station identity
Phone number	10	3	Phone number
Takepicure	15	2	Photographs
Mediarecord	15	1	Media data
Internet	15 + x	$\max(y_i, y_j, \ldots, y_k)$	x: score that sensitive information represents; y: maximum weight in all kinds of sensitive information

We set the scoring range of apps:

$$1 \quad safe: \quad 0 < score <= 25 \tag{3}$$

$$2 \quad low\ risk: 25 < score <= 80 \tag{4}$$

$$3 \quad medium\ risk: 80 < score <= 125 \tag{5}$$

$$4 \quad high\ risk: score > 125 \tag{6}$$

For example, App A obtained the location position, the phone numbers of user, and also invoked network interface:

NSURL Connection::send Synchronous Request:returning Response, and data packets contains location and phone number, so the score of A is:

$$\begin{aligned} \sum_{i=1}^{n=3} score &= factor1 * weight1 + factor2 * weight2 + ... \\ &+ factorn * weightn + ScoreInternet \\ &= 5 * 2 + 10 * 3 + (5 + 5 + 10) * 3 \\ &= 100 \end{aligned} \tag{7}$$

Thus, the security level of A is medium risk.

References

1. Al-Hadadi, M., Al Shidhani, A.: Smartphone security awareness: Time to act. In: Current Trends in Information Technology (CTIT) (2013)
2. http://www.rdacorp.com/2012/08/mobile-application-developmentsecurity/
3. Li, Q., Clark, G.: Mobile security: A look ahead, security & privacy. IEEE **11**(1), 78–81 (2013)
4. Ahmad, N., Musa, M.S.: Comparison between android and ios operating system in terms of security
5. Charlie, M., Dion, B., Stefan, E., Vincenzo, I., Ralf-Philip, W.: iOS Hacker's Handbook
6. http://www.apple.com/iphone/
7. saurik: http://iphonedevwiki.net/index.php/mobilesubstrate