Volume 4, Issue 8, August 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Review of Signature Generation for Private Information Leakage in Android Applications

Prof. D. N. Rewadkar
Dept.of Computer Engineering,
RMD Singed College of Engineering,
Pune, India

Pavitra Mangesh Ratnaparkhi
Dept.of Computer Engineering.
RMD Sinhgad College Engineering,
Pune, India

Abstract: Currently, the number of android phones (smart phone) as well as its applications has been increased. Particularly, there are many applications that are "free" to the user, but they are depend on advertisement services for their expenses .Such applications include an advertisement module that can collect a user's sensitive information from library and broadcast it across the network. Such information is used by some advertisements which are interested in user's personnel data, and users' behavioral information. Users have to accept this business model as users want to use free application services, but in most cases the applications do not require the user's acknowledgment in order to send out susceptible information .Therefore, such applications' activities becomes an incursion of privacy. To enable users to control the diffusion of their clandestine information, there is technique using a new clustering method based on the HTTP packet sender, HTTP packet destination and content distances, creates signatures from the grouping result and uses them to distinguish sensitive information outflow from Android applications .The system does not require an Android framework alteration or any special rights towards the resources.

Keywords: Mobile Security, Android phones, Privacy, Dalvik VM, Android framework.

I. INTRODUCATION TO CLUSTERING METHOD

Smartphone's rapid growth has lead to a renaissance for mobile services. Applications support a variety of financial, and social, enterprise services[3].Application markets provides onclick access to thousands of paid and free applications. With the rising esteem of smartphones and tablets, progress for operating system of mobile device (predominantly for Apple's iOS and Google's phone, motorola, which are the most admired choices) has considerably increased, particularly the growth of applications for online market(library of applications) for instance the AppStore and Google Play. Normally we are interested in free application which comes with an Advertisement Modules [1].

A smartphone keeps a variety of private information, such as location information, contents of the user's address book, and the unique device identifier with the intention of decoupling the features of the device (ie. banking through mobile, network right to use, the camera, susceptible information), and thus preserve protection, Android provides a framework which requires applications to have precise authorization accessing confidential resources. Nevertheless, the Android authorization framework does not totally look after the user's sensitive information applications or advertises with certain authorization combinations can transmit the user's sensitive information to the remote servers using the network [1].While this information is generally used for besieged marketing, it can also be exposed and used by malicious parties without the real user's consideration [3], [5].Various methods employing tracking information flow and privilege partition have been used to address this problem [3]. These approaches require wide spread alterations to the Android framework. It is adequate to force applications to report users of information handling details, therefore enabling them to dynamically control the handling of sensitive information. Objective of this approach is practical method that can identify when applications leak sensitive information without an Android framework alteration, and to create a system where users can easily manage the transmission of sensitive information by applications, thus reducing the possible disobediences to privacy .Clustering method which uses selected HTTP packets to create signatures that can correctly categorize new HTTP packets containing sensitive information. Free software that permits sensitive information leakage is not malware (and therefore is not noticed by anti-virus software) but still presents a threat to the user's privacy [6]. Application of clustering to a sample of the suspicious data (the packets holding sensitive information) to create signatures, and re-applied these signatures to the entire dataset [1]. Created signatures have adequate accuracy for identifying sensitive information diffusions.

Main parts of the system to be:

1) HTTP packet distance Clustering method that recognizes the similarity between the two network Android application packets.

2) Signature generation from the application of clustering method that can identify sensitive information outflow without alteration of the Android framework and it is fast.

In Section II, Android Architecture (Android API) and Android Application's Authorizations structure is shown. In Section III, problem description of android application permissions is shown. In Section IV, Algorithms for the HTTP packet clustering and signature generation are given. In V, Conclusion and future work.

II. BACKGROUND

A.Android Architecture Framework

Figure 1 shows the Android architecture overview , which consists of following parts 1) Linux kernel, 2) Middleware, 3) Android applications, and 4) sensitive information consist of Address Book, Location, Mail, and Phone State .

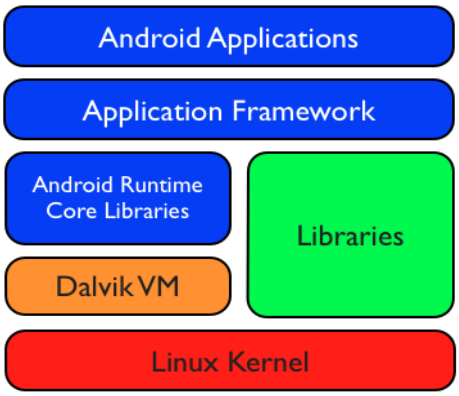

Figure 1: Android Architecture

Middleware consist of the Library framework, one Dalvik Virtual Machine (DVM) for each application and binder in android smartphone.

The Linux kernel provides some basic features for the above layers: 1) Process management, 2) file system management and 3) network services. Dalvik Virtual Machine are use to run applications and the Binder, which supports IPC and checks an application's authorization list when it tries to access sensitive information of user via Library. On Android phone,Each application installed on android phone have a unique Linux UID and resources have Linux UID specific authorizations. This pattern is called sandboxing.

B. Android Application's Authorizations structure

Android framework provides facilities to carry authorization permissions. For accessing resources on Android, a specific set of authorizations needed by an application which link to the resources. For example, The READ_PHONE_STATE permission allows an application to access sensitive and private information which consist of the unique device identifier and line number on the device. The INTERNET authorization sanctions an application to connect to any outer network. There are nearly 125 privileges authorizations defined by Android API Level 15 [2], [10]. When an application accesses a sensitive service object, the Binder manage the application's request with the help of reference monitor [1]. The Binder verifies that the application has the appropriate authorizations to connect to the requested service [1].

III. PROBLEM DESCRIPTION OF ANDROID APPLICATION PERMISSIONS

In this section, The Point covered is how some arrangements of application authorizations can allow a desecration of user privacy.

A. Android Application Request Authorization Pattern:

Normally many free applications require the INTERNET permission which is used to connect to Network and some combination of sensitive information authorizations which consist of LOCATION, READ_PHONE_STATE and READ_CONTACTS [1]. Only some loyal applications from trusted provider needs single INTERNET applications. Those free applications are able to access sensitive information on the android phone and send information gathered to outside servers, this activity happens without user confirmation, putting the user's privacy at risk. In current Android structure, an application requests for permissions once at the time of installation. After the installation of this Application, It sends the users private information to advertisement modules which is totally unclear to the user. User is totally unaware of this thing so there has no way of resolving if sensitive information is present in his network traffic [1].

B. Android Application Traffic:

Commonly Advertises are interested in UDID's [4]. The types of UDIDs consist of the Android ID, the International Mobile Subscriber Identity (IMSI), the International Mobile Equipment Identity (IMEI), and the SIM Serial ID. Some modules compute UDID's hash with a cryptographic hash function at the time of transmission [5]. These UDIDs are fixed and linked to a user's real name and bank account (private well as very sensitive information). UDIDs are hard to change or remove as they are absolute, so if these UDID's are licked by any way then it may be harmful for user's privacy. Most common hosts and the number of applications that send information to each destination domain. Many applications send HTTP packets to the same destinations ,then there may have high possibility that they are sending sensitive information of user to advertisement modules [1].

IV. SYSTEM DESCRIPTION

Without an Android general architecture alteration, this System identifies network's doubtful behavior especially sending of user's private information to outer server by applications. As the system is not modifying basic architecture of Android, the system does not require any special permissions, that why the system is practical, trivial and simple to use for users .Proposed system collects network traffic and create signatures from the clustering of network traffic [1]. If sensitive information is sent in unencrypted format via the network, it is a very simple to identify such transmission. It is also useful against encrypted traffic that uses the same encryption key over a variety of modules or applies a cryptographic hash function to sensitive information [1].

Normally free applications send out sensitive information to external servers. Main causes for this is that developers build an advertisement module into the free version for their revenue. So as to collect private information of the device usage and to provide targeted advertisements for users, advertisement modules take advantage of their ability to access sensitive information [1] and Android framework is not able to detect all sensitive leakages from installed applications.

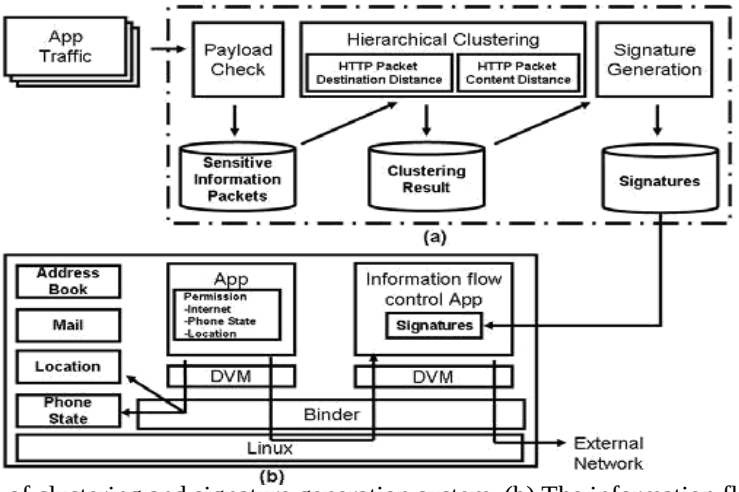

Figure 2 (a) The architecture of clustering and signature generation system. (b) The information flow control application that uses the signatures created by (a), in [1].

A.System Overview

System consists of two parts 1) a separate server collects application's packet traffic, data clustering and signatures generation from clustered data. 2) Flow control application for information on the user's device (shown in Figure 2b) extracts signatures from the servers and handles the transmission of other applications' network traffic.

The server creates signatures by the following process. 1) It creates a payload check, which separates application network traffic into two groups: one containing packets with sensitive information, and the other not containing sensitive information [1]. 2) The server forms cluster holding sensitive information based on packet destination distance and contents distance, and builds a set of signatures using combination of signatures [1]. Using the HTTP packet distance allows differentiating tendency and distributions of HTTP packets. Accordingly sensitive information packet will be clustered with other sensitive information packet recursively, generating a useful signature from those packets. For creating such practical signatures, distance is defined to take in both contents of packet and packet destination. This wider definition results into packets transmitted to similar server to be clustered together, creating advertisement module specific signatures. The information flow control application identifies network traffic using the Android API and inspects sensitive information leakage using proposed server created signatures. There is no need of android framework modification.

B. HTTP Packet Destination Distance

In [1], The HTTP packet destination distances are evaluated by the packets' destination IP addresses, port numbers, and HTTP host domains. Given two HTTP packets *px* and *py*, the HTTP packet destination distance is calculated as

$ddst(px, py) = dip(px, py) + dport(px, py) + dhost(px, py).$

Let HTTP packet pn destination be defined as

$pn = \{ipn, portn, hostn\}$, where *ipn* is a destination IPv4 address, *portn* is the port number on *pn* destination, *hostn* is HTTP host. The distances given in above equation are defined as follows:

1) *Destination IP Address Distance*

In [1], the destination distance IP address on packets *px, py* as

$dip(px, py) = lmatch(ipx, ipy)/32 \in [0, 1]$

where *lmatch* is a function returns a number of common upper bits in two IP addresses.

2) *Port Number Distance*

In [1], Port numbers distance returns Boolean value. Normally, specific port numbers are reserved for services. The port number distance on packets *px, py* as

$dport(px, py) = match(portx, porty) \in \{0, 1\}$
,where *match* is a function returns 1 on matching ports, and 0 on different ports.

3) *HTTP Host Distance*
In [1], The distance between HTTP host domains can be calculated using the generalization method to resolve their edit distance. The HTTP Host distance on packets *px, py* as

$$dhost(px, py) = \frac{ed(hostx, hosty)}{max(len(hostx), len(hosty))} \in [0,1]$$

where *ed* is a function which returns an edit distance result, *len* is a function which returns a length of character strings, and max is a function which returns the greater of its two input values.

C. HTTP Packet Content Distance
In [1],The HTTP packet content distance is calculated using the request-line, cookie, and message-body fields of the HTTP header. Given two HTTP packets *px* and *py,* The HTTP content distance *dheader(px, py)* as
dheader(px,py)=drline(px,py)+dcookie(px, py)+dbody(px, py).
Let HTTP packet *pn* contents be defined as *pn = {rlinen, cookien, bodyn},* where *rlinen* is request-line, *cookien* is cookie, *bodyn* is message-body. These contents are in the form of character or binary strings. For correct distance calculation, application of the normalized compression distance (NCD) [10] algorithm is suitable.

D. Hierarchical Clustering of packets
In [1], Hierarchical clustering uses group averages for iterative calculation and calculates the proximity of clusters with HTTP packet destination distance and HTTP packet content distance. It then allots a cluster to each HTTP packet, and iteratively forms new clusters from the nearest distance of HTTP packet pairs until there is only one cluster. Given two HTTP packets *px* and *py*, HTTP packet distance
dpkt(px, py) = ddst(px, py) + dheader(px, py)
ddst and *dheader* are already calculated in above sections IV-B and C respectively.

E. Signature Generation
In [1], Combination of signature set created from hierarchical clustering results, which is a HTTP packet group. A conjunction signature set consist of the invariant tokens that describe the longest common substrings. Every Signature symbolize a feature of the cluster. Given a dataset of N HTTP packets H = {*pi*}i=1..N and a subset *Pj,j*=1..M ⊂ H used to create HTTP packet group C, which has the nesting structure characteristic of clusters, conjunction signature set created using the following process:
1) Select the top of cluster *Ci ∈ C.*
2) Calculate a signature *Si* as longest common strings of HTTP contents in *Ci.*
3) Remove *Ci* from *C* and repeat for all clusters in C.

V. CONCLUSION AND FUTURE WORK
In this system, Signatures can be stored in the Database to identify packets which carries sensitive information of the user. New thing is when some packet will goes out from the mobile, all signatures will be stored and next time these stored signatures will be compared to the outgoing packets from network.
In [7],[6] ,Technique have shown that many Android applications require permissions for sensitive information access and network features, and that among them are applications that connect to many outside servers and send sensitive information of user without the user's acknowledgment. Clustering method using HTTP packet distances which include both the distance between HTTP packet destinations and between HTTP packet contents which is prominent measure of identifying suspicious information enclosed in application traffic packets.
In [8], Other approaches to prevent sensitive information leakage include taint tracking and permission framework modifications. Taint tracking can also accurately detect sensitive information leakage and control the information flow of applications but those approaches require a little modification in Android framework.
In [9], permissions of applications in android phone are modified by applications themselves because android framework is not much strong to prohibit them and user information is stolen by them.
There are many methods for pattern and signature generation from packet clustering results, but those methods need Android framework modification. Advertisement services are widely accepted among application developers for their revenue. However it is still important to investigate the behavior of free application for security and privacy.

References
[1] Signature Generation for Sensitive Information Leakage in Android Applications Hiroki Kuzuno 1, Satoshi Tonami 2 Intelligent Systems Laboratory, SECOM, Tokyo, Japankuzuno@secom.co.jp 2s-tonami@secom.co.jp
[2] P. Hornyack, S. Han, J. Jung, S. Schechter, and D. Wetherall, "These aren't the droids you're looking for: Retrofitting android to protect data from imperious applications," in 18th ACM Conference on Computer and Communications Security (CCS 2011), Nov. 2011.
[3] W. Enck, D. Octeau, P. McDaniel, and S.Chaudhuri, "A study of android application security," in 20th USENIX

Security Symposium 2011, Aug. 2011.

[4] M. Grace, W. Zhow, X. Jiang, and A.-R. Sadeghi, "Unsafe exposure analysis of mobile in-app advertisements," in 5th ACM
Conference on Security and Privacy in Wireless and Mobile Networks (WiSec 2012), Apr. 2012.

[5] R. Stevens, C. Gibler, and J. Crussell, "Investigating user privacy in android ad libraries," in Mobile Security Technologies (MoST 2012), May 2012.

[6] "Karelog,"Manuscript.Inc,2011, http://karelog.jp/.

[7] ASIAJIN, "Android app karelog lets you spy on your boyfriend remotely," 2011, http://asiajin.com/blog/2011/08/31/android-app-karelog/.

[8] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: An information-flow tracking system for re-altime privacy monitoring on smartphones," in 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2010), Oct. 2010.

[9] A. P. Felt, H. J. Wang, A. Moshchuk, S. Hanna, and E. Chin, "Permission redelegation: Attacks and defenses," in 20th USENIX Security Symposium 2011, Aug. 2011.

[10] J. Jeon, K. K. Micinski, J. A. Vaughan, N. Reddy, Y. Zhu, J. S. Foster, and T. Millstein, "Dr. android and mr. hide: Fine-grained security policies on unmodified android," in CS-TR-5006, Dec. 2011

Author Profile

Pavitra M. Ratnaparkhi received BE degree in Information Technology in 2013 and persuing ME in Computer Engineering from Sinhgad School of Engineering, Warje, Pune.

