IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 3 (Jul. - Aug. 2013), PP 53-58
www.iosrjournals.org

A Framework for Providing Selective Permissions to Android Applications.

Aditi Tripathy[1], Prof.G.P. Potdar[2]
[1,2](Computer Science, Pune Institute of Computer Technology/ University of Pune, India)

Abstract: *The latest version of Android 4.2 doesn't provide selective permissions for apps while they are being downloaded.This doesn't provide flexibility to the user to restrict the usage of their resources by the app. This paper proposes a framework for providing selective permissions in Android Operating System*
Keywords: *Android Operating System , Design , Protection , Security Selective Permissions, Verification .*

I. INTRODUCTION

The core Android Operating System is based on the Linux Kernel. Android applications [3] is most often written in Java programming language and run in the Dalvik Virtual Machine[4]. Applications are installed from a single file within the .apk file extension. Android is a privilege separated Operating System, in which each application runs with a distinct system identity (Linux user id and group id).Parts of the system are also separated into distinct identities. Linux thereby isolates applications from each other and from the system. Additional finer grained security features are provided through a "permission" mechanism that enforces restrictions on the specific operations that a particular process can perform, and per URI permissions for granting ad-hoc access to specific pieces of data .

Applications statically declare the permissions they require, and the Android system prompts the user for consent at the time the application is installed, i.e. all permissions that an app requires are granted at the time it is installed. Before installation the user is shown the list of permissions that the app requests. The user must then choose whether to grant all of the requested permissions and install the app or deny the permissions and cancel the installation.

There is no way to install an application while specifically denying some of the per-missions it has requested. Similarly there is no way to block access to sensitive re-sources after an app has been installed. The only way to block in the latter case is to uninstall the application. There is no way for an application to request additional permissions after it has been installed. If an updated version of an application requires more permissions than the older version, the user must approve the new set of requested permissions.

While the existing security model ensures that an application cannot perform any inappropriate actions, the burden is largely placed on the user to decide exactly what is appropriate for an application. A user must decide without ever having run an app, exactly what actions the app should be allowed to perform. Deciding whether to install a new application that requests a long list of permissions is a daunting process for any user.

1.1 ANDROID APPLICATION COMPONENT

Android application building blocks consists of[8]

Activities: An Activity is, generally, the code for a single, user-focused task. It usually includes displaying a UI to the user, but it does not have to. Some Activities never display UIs. Typically, one of the application's Activities is the entry point to an application.

Services: A Service is a body of code that runs in the background. It can run in its own process, or in the context of another application's process. Other components "bind" to a Service and invoke methods on it via remote procedure calls. An example of a Service is a media player: even when the user quits the media-selection UI, the user probably still intends for music to keep playing. A Service keeps the music going even when the UI has completed.

Content Providers:They manage application data and interact with the SQL databases. They are also preferred means of sharing data across application boundaries.

Intents: It is a powerful interapplication message passing framework. Used to start and stop activities and services, to broadcast message system wide or to an explicit Activity, Service or Broadcast Receiver or to request an action to be performed on a particular piece of data.

Broadcast Receiver: A Broadcast Receiver is an object that is instantiated when an IPC mechanism known as Intent is issued by the operating system or another application. An application may register a receiver for the low battery message, for example, and change its behavior based on that information.

Widgets: A special variation of a Broadcast Receiver, widgets enable you to create dynamic, interactive application components for users to embed on their home screens.

Notifications: They enable you to alert users to application events without stealing focus or interrupting their current activity.

II. RELATED WORK

In Shadow Manifest[7] permission that an app requires are stored by prior execution of the app.Unecessary permissions are stored with a mask which are to be revoked by generating an empty resource when an app requests them.

COPES(Correct Permission Set)[6] is a tool which uses static analysis to extract a table from from Android framework bytecode.This table the set of permissions that an app needs and maps every method of the API to these permissions which are called.So no unnecessary permissions are stored in the table and mapped with API methods.

In Apex[5] users are allowed to specify what an app can access.An extended installer is used to set user policies.

III. PROPOSED FRAMEWORK

3.1 SKETCH OF PROPOSED FRAMEWORK

Fig1 shows the sketch of the proposed framework.Initially the app has to be installed.The permissions that the app requires are stored in the database.We would already be having a permission set stored in the database divided into various categories.The requested permissions will be stored under these categories.

Now the run time permissions of the app will be checked with the initial permissions it has requested for.If they match the app will be provided with the resource.If they do not match the user will be shown a notification warning of malicious behavior.Now the resource will be provided after a time delay.The user can in the meantime stop the app if he considers it to be dangerous.

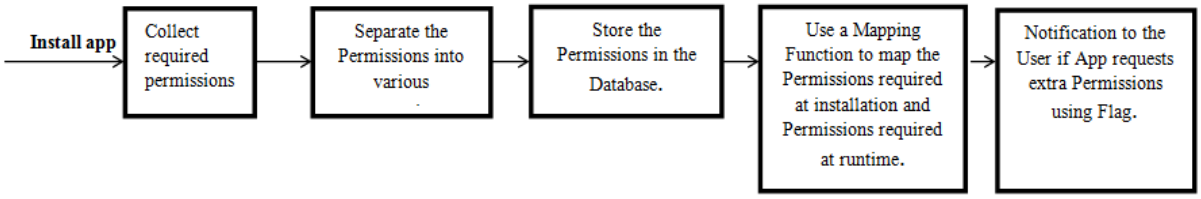


Fig3Sketch of proposed framework.

3.2 Class Diagram Of Proposed Framework

Fig2 shows the class diagram of the proposed framework. There are four major classes in the framework. These are the Collect Permission Class which will extract the required permissions of the application at the time of its installation and divide it into various categories. The StorePermission category class will store the collected permissions under various categories.Then there is MapPermission class which will map the run time permissions requests with the app permissions request at the time of its installation.The notify class will notify the user if the app requests extra permissions at run time.There is a one to one relation between the CollectPermission and the StorePermission Class.That is one CollectPermission can have only one StorePermission and vice-versa.

There is a one to many relation between the NotifyUser and the MapPermission class.That is one NotifyUser class can have one or more than one MapPermission class and vice-versa.

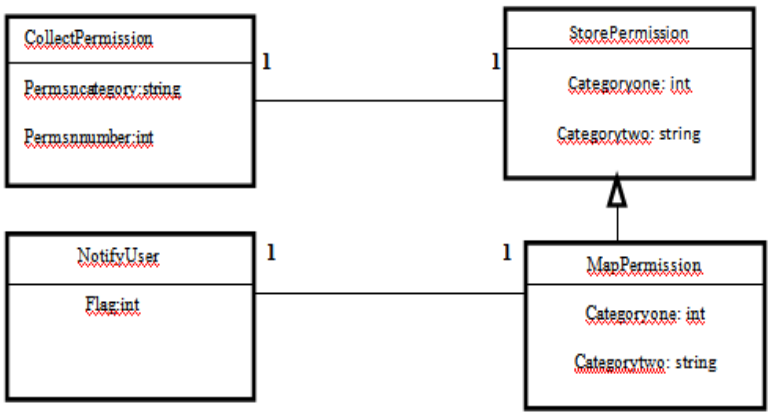


Fig2 Class Diagram of Proposed Framework

3.3. MATHEMATICAL MODEL

A Let S be the proposed system given by

$S= \{I_p, O_p, F_s, S_c, F_c\}$

Where,

.I_p: Input

.O_p: Output

.F_s :Function Set.

.S_C: Success Case.

.F_c: Failure Case.

$I_P = \{P_s, Pu\}$

P_s: Permission Set of the device.

Pu: Permission that an app requires at the time of installation.

$P_s= \{P_1, P2, P3, P4, P5, P6, P7\}$

P_1:Permission that require services that cost money.

$P_1=\{P_c,P_t\}$

$P_c \subseteq P_1 \Leftrightarrow \forall x[x \epsilon P_c \Rightarrow x \epsilon P_1]$

$P_t \subseteq P_1 \Leftrightarrow \forall x[x \epsilon P_t \Rightarrow x \epsilon P1]$

P_c:Permission for calling from device.

P_t:Permission for texting from device.

P_2: Permission that require storage access from your device.

$P_2= \{P_r,P_w,P_d\}$.

$P_r \subseteq P_2 \Leftrightarrow \forall x[x \epsilon P_r \Rightarrow x \epsilon P_2]$

$P_w \subseteq P_2 \Leftrightarrow \forall x[x \epsilon P_w \Rightarrow x \epsilon P_2]$

$P_d \subseteq P_2 \Leftrightarrow \forall x[x \epsilon P_d \Rightarrow x \epsilon P_2]$

P_r: Permission to read from your SD card or internal memory.

P_w: Permission to write from your SD card or internal memory.

P_d: Permission to delete from your SD card or internal memory.

P_3:Permission to access your personal information.

$P_3=\{P_{ct},P_{cd},P_{ac},P_{li},P_{br}\}$

$P_{ct} \subseteq P_3 \Leftrightarrow \forall x[x \epsilon P_{ct} \Rightarrow x \epsilon P_3]$

$P_{cd} \subseteq P_3 \Leftrightarrow \forall x[x \epsilon P_{cd} \Rightarrow x \epsilon P_3]$

$P_{ac} \subseteq P_3 \Leftrightarrow \forall x[x \epsilon P_{ac} \Rightarrow x \epsilon P_3]$

$P_{li} \subseteq P_3 \Leftrightarrow \forall x[x \epsilon P_{li} \Rightarrow x \epsilon P_3]$

$P_{br} \subseteq P_3 \Leftrightarrow \forall x[x \epsilon P_{br} \Rightarrow x \epsilon P_3]$

P_{ct}: Permission to access your contact data.

P_{cd}: Permission to access your calendar data.

P_{ac}: Permission to access your accounts and sync data.

P_{li}: Permission to access your language and input data.

P_{br}: Permission to access your backup and reset data.

P_4: Permission to read phone state and identity.

$P_4=\{x|x \text{ is phone state and identity}\}$

P_5: Permission to access your location services.

$P_5=\{P_{gp},P_{wf}\}$

$P_{gp} \subseteq P_5 \Leftrightarrow \forall x[x \epsilon P_{gp} \Rightarrow x \epsilon P_5]$

$P_{wf} \subseteq P_5 \Leftrightarrow \forall x[x \epsilon P_{wf} \Rightarrow x \epsilon P_5]$

P_{gp}: Permission to determine your location by GPS.

P_{wf}: Permission to determine your location by Wi-Fi or Mobile Network.

P_6: Permission to access your wireless and network resources.

$P_6=\{P_{wi},P_{bl},P_{du}\}$

$P_{wi} \subseteq P_6 \Leftrightarrow \forall x[x \epsilon P_{wi} \Rightarrow x \epsilon P_6]$

$P_{bl} \subseteq P_6 \Leftrightarrow \forall x[x \epsilon P_{bl} \Rightarrow x \epsilon P_6]$

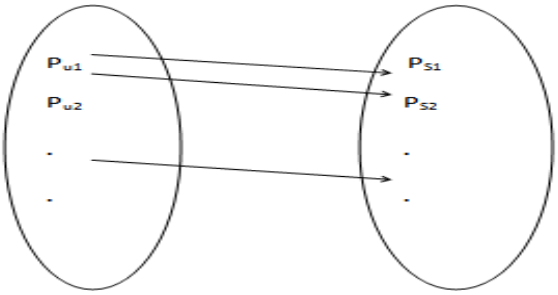
$P_{du} \subseteq P_6 \Leftrightarrow \forall x[x \epsilon P_{du} \Rightarrow x \epsilon P_6]$

P_{wi}: Permission to access your WiFi network .
P_{bl}: Permission to access your bluetooth network .
P_{du}: Permission to access your data usage.

P_7: Permission to access your other device resources.

$P_7=\{P_{so},P_{wa},P_{di},P_{si},P_{bt}\}$
$P_{so} \subseteq P_7 \Leftrightarrow \forall x[x \epsilon P_{so} \Rightarrow x \epsilon P_7]$
$P_{wa} \subseteq P_7 \Leftrightarrow \forall x[x \epsilon P_{wa} \Rightarrow x \epsilon P_7]$
$P_{di} \subseteq P_7 \Leftrightarrow \forall x[x \epsilon P_{di} \Rightarrow x \epsilon P_7]$
$P_{si} \subseteq P_7 \Leftrightarrow \forall x[x \epsilon P_{si} \Rightarrow x \epsilon P_7]$
$P_{bt} \subseteq P_7 \Leftrightarrow \forall x[x \epsilon P_{bt} \Rightarrow x \epsilon P_7]$

P_{so}: Permission to access your device sound.
P_{wa}: Permission to access your device wallpaper.
P_{di}: Permission to access your device display.
P_{si}: Permission to access your device
simcard
P_{bt}: Permission to access your device
Battery.



App requested Permission Permission set of the device.

Function Set:
F_s= {f0,f1, f2, f3, f4, f5, f6, f7}
f0: Pu→Ps

f1:P1→a
a ϵ {P_c U P_t}
f2:P2→b
b ϵ {P_r U P_w U P_d}

f3:P3→c
c ϵ {P_{ct} U P_{cd} U P_{ac} U P_{li} U P_{br}}

f4:P4→d
d ϵ{x}

f5:P5→e
e ϵ{P_{gp} U P_{wf}}

f6:P6→f
f ϵ {P_{wi} U P_{bl} U P_{du}}

f7:P7→g
g ϵ {$P_{so,}$ Pwa,Pdi,Psi,Pbt}

Output:
O_P={N,Z,D}

N:Notification to user.
$N=\{n_1,n_2\}$
$\{n_1, n_2\} \epsilon$ R
R=True(notify), False(do not notify)

Z: Flag for notifying faulty applications
Z= {0, 1}
0 ⇒ no faulty apps
1 ⇒ faulty app.
If $Z \epsilon 1 \Rightarrow N$

D:Delay access to resources.
If $N \epsilon n_1 \Rightarrow D$.

Initial Condition:
$I_P \neq \phi$

Success case:
$S_c=S_1 \cap S_2$
$S_1 \Rightarrow$Malicious app detected successfully and resource access is delayed.
$S_2 \Rightarrow$User is notified at the right time.

Failure case:
$F_c=F_1 \cap F_2$
$O_P=\phi$
$F_1 \Rightarrow$Malicious app not detected.
$F_2 \Rightarrow$User not notified.

3.4 Algorithm
Algorithm PermissionMapping (P_u,P_r)

```
{
   x: Application;

if (Pu(x)==Pr(x)) //Pu:Initial Permissions.
                  // Pr: Runtime Permissions.
{
then grant resource to x;
return x;
}

else

{
Notify user and delay grant of resource to x;
return    x+delay;
}
```

IV. Conclusion and Future Work

In this paper we have presented a framework for enabling the user to provide selective permissions in terms of selective resource access to the app during runtime.Here we have considered malicious behavior as requesting for extra permissions at run time as compared to the permissions asked at installation time.

The future work would include recognizing extra permissions as fully malicious or whether it is a genuine request by the application and then enabling the user to decide the next course of action.

References

Journal Papers

[1] Nwokedi Idika,Aditya P. Mathur,A Survey of Malware Detection Techniques,.http://www.serc.net.,Last Acessed:November 1,2012.

[2] Mahinthan Chandramohan,Hee Beng Kuan Tan.Detection of Mobile Malware in the wild.IEEE Computer Magazine Sep2012.

[3] Android open sourceproject, Application Sandbox.,http://source.android.com/tech/security/index.html

[4] Android open source project, Elements of Applications, Interprocess Communication, http://source.android.com/tech/security/index.html

[5] Mohammad Nauman and Sohail Khan. Design and Implementation of a Fine-grained Resource Usage Model for the Android Platform. The International Arab Journal of Information Technology, Vol.8, No.4, Oct 2011

[6] Alexandre Bartel,Jacques Klein,Martin Monperrous.Automatically Securing Permission-Based Software by Reducing the Attack Surface:An Application to Android.

[7] Lakhmi Priya Sekar,Vinitha Reddy Gankidi, Selvakumar Subramanian.,Avoidance of Security Breach through Selective Permissions in Android Operating System.ACM SIGSOFT Software Engineering Notes,September 2012 Volume 37 Number 5.

Books

[8] Reto Meier. *Professional Android 4 Application Development*.Wiley India Publication.