# Design and implementation of lock function in Android application

Tang Hao [1, a], Wang Yongchao [2, b]

[1]Synergy Innovation Institute of GDUT, Heyuan, 517000, China

[2]Synergy Innovation Institute of GDUT, Heyuan, 517000, China

[a]email: 1039829238@qq.com

**Keywords:** Application; lock; application list

**Abstract.** For the protection of user’s privacy information, when the mobile phone is lost in the hands of others or when others use their own mobile phones, while do not want their own privacy or security information is opened by others, this paper studies and implements an application lock function. Through managing the application, the user can put the applications that they want to protect into the application lock and set a password, only enter the correct password to access, thereby enhanced the user's privacy information security.

## Introduction

While Android inherits the security of the Linux kernel, and has some of its own extended security mechanisms to protect the security of the system and the application, and also achieve the security of the system resources and sensitive data through the authority mechanism. But in the face of complex environment, personal privacy information is still faced with a huge threat, which mainly comes from the users’ mobile phone may be lost to the hands of others, so the sensitive information of their own privacy can easily be seen by others; in addition when others use their own mobile phones, the information users wanted to protect may be intentionally or unintentionally opened to operate and browse by others, such as PayPal, photo album, address book and other related to the user's privacy, this will bring damage for users, so this paper presents the application lock design.

## Function analysis

The first time access to the application lock will require the user to set a password, after setting it will jump to the application list Activity interface, the interface has unlocked application list and locked application list. In unlocked application list, it shows all the pre-installed software and third party applications in the system and they can be put into locked application list. Similarly, locked application list shows all the applications that have been locked by users, and you can also click them to unlock the application, at the same time, the application will appear in unlocked application list and disappear from locked application list. So it is concluded that the following functions need to be realized：

a. Set a password to application lock
b. Choose the applications need to be locked
c. Cancel the application lock

## Password setting

In Android devices, the common lock mode is in the form of characters, graphics, face recognition and PIN code, but they are the same principle of through certain encryption process, the result is stored in the database file. The input password is matched with the password in database. When the first access to the application lock is required to set a password, the main code of setting password is as followed:

```
public void setPassWord(final String pN){
    newAlertDialog.Builder(this).setTitle("setting password ").
    setView(setPwd).setButton("confirm", new DialogInterface.OnClickListener(){
    public void onClick(DialogInterface d, int w) {
    if(pwd1==pwd2){
    if(pwd1==NULL||pwd2==NULL){
    //password cannot be NULL
    ...
    }else{
    ...
    Toast.makeText(MainActivity.this, "password setting successfully", Toast.LENGTH
    _SHORT).show();
    Toast.makeText(MainActivity.this, "lock successfully", Toast.LENGTH_SHORT).show();
      ......
    }
    }else{
    //input password
    String inputText = etInputPwd.getText().toString().trim();
    //get the password
    preferences = getSharedPreferences("passWord", MODE_PRIVATE);
    passWord = preferences.getString("pwd", "");
    //verify password
    ...
    }
    }
 }
}
```

## Design of application lock

As to the implementation process of application lock, the first step is to lock the application; and put the locked application into database. When the user open the locked application, there will be a service to monitor the application on the top of the stack, and match the current stack top application with database encryption applications, if match successfully, a password input interface will pop-up for the user to enter a password to unlock, and then can access applications if unlock successfully. Application lock theory is as the Fig.1 shown.

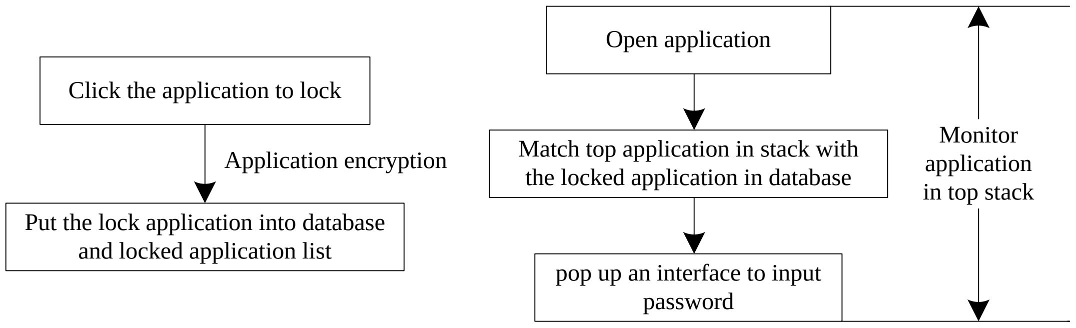
Click the application to lock
Application encryption
Put the lock application into database and locked application list
Open application
Match top application in stack with the locked application in database
pop up an interface to input password
Monitor application in top stack

Fig.1. Implementation of lock function in Android application

a. Put the lock applications into database

In unlocked application listView, the user can click element in listView, then listView.onClickListener interface in listView will monitor the events and use if(ls.onClick()) statements to determine the user's operation, after that the Handler mechanism will distribute the event to doAddLockApp() method and put the applications into locked application list, also call the insertLockAppBean() method in database to put the applications into locked application list database.

In the same way, in locked application list, click the element in listView, then listView.onClickListener interface in listView will monitor the events and use if(ls.onClick()) statements to determine the user's operation, after that the Handler mechanism will distribute the event to doAddUnlockApp() method and put the applications into unlocked application list, also call the insertUnlockAppBean() method in database to put the applications into unlocked application list database.

b. Open service to monitor the top of stack

When the user opens an application, it needs to pop up an interface to input password, so it must know when user click the application. Therefore, a service to monitor application in top of stack is needed, it will scan top Activity in task, then it must get the task firstly, and getRunningTasks() method will return a sequential List, and first is most recent, so we only need to take out the first one, then get the top activity which is package type in the task. As a result, we can get the package name and class name, the main code is as followed:

```
//get the running task
mActivityManager = (ActivityManager) context.getSystemService("activity");
//get the top Activity in task
ComponentName topActivity= mActivityManager.getRunningTasks(1).get(0).topActivity;
// Get the package name for the Activity to run
String runningpackageName=  topActivity.getPackageName();
```

But when an application is open, the system will not send broadcast, cannot monitor directly, so the paper will adopt regular scanning strategy of checking task every second. Sometimes open an application and unlock it, then press Home button to return to the desktop, and then open the application again, it will pop up the unlock interface, so it is necessary to solve the BUG of appearing verification page repeatedly and monitor whether the package is locked when open the application.

```
if(handleDB.find(runningpackageName)){
        runningApp = runningpackageName;
        // to solve the BUG of appearing verification page repeatedly：
        //if runningApp.equals(lastRunningApp)=true, it indicates that
        //the current stack running program has been unlocked
        if((runningApp.equals(lastRunningApp)) == false){
        intentLockAppActivity.putExtra("packageName", runningpackageName);
        intentLockAppActivity.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        startActivity(intentLockAppActivity);
            }
}
```

c.Monitor the locked application and pop up verification page

The main code is as followed:

```
String input = etInputPwd.getText().toString().trim();
preferences = getSharedPreferences("passWord", MODE_PRIVATE);
```

```
passWord = preferences.getString("pwd", "");
if(TextUtils.isEmpty(input))
{
Toast.makeText(this, "password cannot be NULL", Toast.LENGTH_SHORT).show();
}
else if(passWord.equals(input))
{
    // Assignment to solve the BUG of appearing verification page repeatedly
WatchAppService.lastRunningApp = WatchAppService.runningApp;
finish();
}
else
{
Toast.makeText(this, "password mistake", Toast.LENGTH_SHORT).show();
etInputPwd.setText("");              //    set NULL
}
```

d.Add the following configuration information in Androidmanifest.xml file:

```
<serviceandroid:name="com.xh.service.WatchAppService"
  android:persistent="true"></service>
```

Therein, android:persistent="true" set the service as a system service to prevent the system from accidental recovery.

## Conclusion

This paper studies and implements an application lock function. Through managing the application, the user can put the applications that they want to protect into the application lock and set a password, only enter the correct password to access, thereby enhanced the user's privacy information security.

## References

[1] W. Enck, M. Ongtang, P. McDaniel. Understanding Android Security[J]. Security &Privacy, IEEE, 2009,7(1):50-57.

[2] Jeff Six. Application Security for the Android Platform [M]. Sebastopol:O’Relly, 2011.

[3] Haifeng Li, Haiyun Ma, Yanwen Xu. Modern cryptography theory and application [M]. Beijing: National Defence Industry Press, 2013.

[4] Joerg Christian Wolf, Phil Hall, Paul Robinson, Phil Culverhouse. Bioloid based Humanoid Soccer Robot Design, 2007.

[5] Haifeng Li, Haiyun Ma, Yanwen Xu. Modern cryptography theory and application [M]. Beijing: National Defence Industry Press, 2013.