Huawei's Undocumented APIs — A Backdoor to Reinstall Google Services

John Wu
Dec 25 · 6 min read ★

Image from https://thehackernews.com/2016/11/hacking-android-smartphone.html

Update: The "LZPlay" website and download links are no longer accessible. Even if you grabbed the APK before it was gone, it no longer works, as the special certificate required to access the "backdoor" is either revoked by the developer or Huawei. In addition, existing devices that used LZPlay to install GMS no longer passes full SafetyNet Attestation, rendering many apps and services unusable, such as Google Pay and many games.

Right off the bat, here's the TL;DR

> *certificate from Huawei, was granted privileges nowhere to be found on standard Android systems.*

. . .

Ever since news broke out that Huawei's latest flagship smartphone would not be allowed to ship with Google services due to the U.S. trade ban (source: Reuters), people were curious about the impact to sales, and how the Chinese tech giant will react.

After the unveiling event, media got their hands on the "forbidden fruit", and numerous reviews flood the Internet. It doesn't take long before someone found a way to install Google Services on their units (source: 9to5Google), and apparently even Google Pay works. All you need to do is to download and install an APK from https://www.lzplay.net/, follow the instructions in the app, and things are all set.

This sounds too good to be true, doesn't it?

For those who are familiar with Chinese Android devices, sideloading GMS (Google Mobile Service) is nothing foreign. It is very common for Chinese OEMs to release "GMS Installers" so people who travel abroad can install GMS manually.

> *Well, everything seems nice and cool; this "LZPlay" app is just yet another GMS installer, why are you writing this article?*

The way most "GMS Installers" work is that they automatically install a suite of Google APKs. In fact, users can simply just download these APKs individually and sideload them themselves. No magic occurs here. However, this only works if the device is already using a Google licensed system image.

On Android, system apps and user installed apps are treated differently, with the former given additional permissions. Some GMS packages have to be installed as system apps because they require privileged permissions to function properly. As

Android allows system apps to be upgraded by the user, either via Play Store or manual sideloads, as long as the update is signed with the same key as the original one in the system. The signature verification is important, as this prevents attackers from distributing malicious updates. The aforementioned GMS "stubs" are mere placeholders in the system and provide no functionality other than paving the way to be "activated". These stubs are signed by Google for it to be compatible with actual GMS APKs.

· · ·

When I first learned that GMS can be installed on the Mate 30 Pro, I was very surprised:

"*Wait a minute, does that mean either Google is sneaking the stubs to Huawei, or Huawei is blatantly stealing Google's stub binaries?*"

Feeling that either case would be a very "big deal", I asked my friends over at XDA-Developers for some details. The answer shocked me once again: *no stubs can be found in the system*! This means that there is ***magic*** in the "LZPlay" app. I grabbed the APK and immediately used APKTool to do some investigation, and I found something interesting in `AndroidManifest.xml`

```
<uses-permission android:name="com.huawei.permission.sec.MDM_APP_MANAGEMENT"/>
<uses-permission android:name="com.huawei.permission.sec.MDM_INSTALL_SYS_APP"/>
<uses-permission android:name="com.huawei.permission.sec.MDM_INSTALL_UNDETACHABLE_APP"/>
<uses-permission android:name="com.huawei.systemmanager.permission.ACCESS_INTERFACE"/>
```

huawei_permissions hosted with ♡ by GitHub view raw

These doesn't look anything standard, and the permissions sound scary!

After some searching, I eventually stumbled upon a developer documentation for "Huawei Security Authorization SDK" (source: Huawei, in Chinese only). In a nutshell, Huawei has its own set of APIs for mobile device management (MDM), which is often used in enterprises to manage employee devices. Standard Android has its own Device

> *Android Enterprise: Full Feature List*
> *Huawei Security Authorization SDK: API Reference (English PDF)*
> *Huawei Security Authorization SDK: API Reference (Chinese PDF)*

In a quick glance, Huawei's APIs provide more fine grained control over the device, but still all the features listed are reasonable in the sense of MDM. However, 2 of the permissions I listed above are not documented anywhere, which is apparently where the magic of "LZPlay" lies in.

```
<uses-permission android:name="com.huawei.permission.sec.MDM_INSTALL_SYS_APP"/>
<uses-permission android:name="com.huawei.permission.sec.MDM_INSTALL_UNDETACHABLE_APP"/>
```

undocumented_permissions hosted with ♡ by GitHub view raw

These permissions are not documented in the API reference

For some reason, Huawei has undocumented MDM APIs that allow apps to **install system apps** and **install undetachable apps**. It is a well-known trick among Android enthusiasts to "flash an app into system" to unleash system privileges for some specific app; however, in this case it is certainly not the same thing because **a.** the bootloader is locked and Android Verified Boot is enforced; **b.** Huawei format their system/vendor/product partitions as EROFS, a read-only, compressed filesystem. This means the system framework in Huawei's OS has a "backdoor" that allows permitted apps to flag some user apps as system apps despite the fact that it does not actually exist on any read-only partitions.

According to the all-in-Chinese documentation, 3rd party developers/companies are required to sign legal agreements and send them to Huawei in order to gain access to the SDK. For each project, the developer will have to submit a request, along with justification, a list of the permissions willing to be granted. In addition, the APK binary for each release has to be uploaded to Huawei for further examination, which can then finally be signed with Huawei's special key.

. . .

to install Google Services on a non licensed device, and it sounds very sketchy to me, but I'm no lawyer so I have absolutely no idea of its legality.

But even if it is legal, this backdoor should never exist in the first place from a security standpoint. There is a reason why system apps are allowed to have additional privileges: they exist on a cryptographically verified read-only partition. Despite the fact that the certificate to escalate a user app to system app is gate-kept by a trusted(?) party, Huawei, as long as things are stored on a writable partition (userdata), it is susceptible to malicious tampering, and should not be treated the same.

The "LZPay" app is obfuscated/encrypted by QiHoo Jiagu (奇虎加固), and is non trivial to reverse engineer. The more interesting part should lay in Huawei's system image though, but I do not have a Huawei device in my hands to do further analysis (and I'm pretty much done with this at this point). Maybe there are more hidden gems, more unthinkable permissions to be discovered, who knows?

This undocumented API is not the "OMG Huawei is spying on us OMG" kind of backdoor many media might wish to exist. It is protected behind rigorous verification on Huawei's side and requires user interaction to allow the permission to be granted.

Nevertheless, only Huawei knows the intent to create such API and allow the existence of "LZPlay", and it is up to anyone's imagination.

)

Android Huawei Security Backdoor Google