FULL DISCLOSURE: The com.proxgrind.chameleon Android app is malware??? #271

Closed **maxieds** opened this issue on Aug 23, 2020 · 12 comments

maxieds commented on Aug 23, 2020 Contributor

It has come to me that BugTraq, the late once formerly great, is defunct as of this year. Sadly that would have been the best outlet for this sort of report. This is, I suppose, the next best place to voice my position and discovery of this abhorrent attitude on the part of the Proxgrind developers. This report comes well after the BugTraq-era window I communicated to the vendor in light of the fact that that security following conscience is no longer centralized. Their complete disregard for users, and knowledge that this has gone on is so disgustingly apparent that it's harming more folks than not by not reporting it.

Please clone my summary repository, and make sure it cannot disappear, by running the following command at your terminal of choice:

```
git clone https://github.com/maxieds/ChameleonProxgrindAndroid-FullDisclosure.git
```

please join me in a lively follow-up discussion about what should be done about this?

maxieds commented on Aug 23, 2020 Contributor Author

@quantum-x; Gentlemen, anything you want to add to the discussion given your might in making sure these devices ended up in my possession this year? Really, this is a big f*cking deal, boys!!

david-oswald commented on Aug 24, 2020 Collaborator

Hi **@maxieds**

I'm also not sure what is the best place to report this kind of things, maybe try reaching out on the discord channel to them?

The issue you describe seems more like dynamic code loading to obfuscate their app - but yeah it could of course be misused to load arbitrary code.

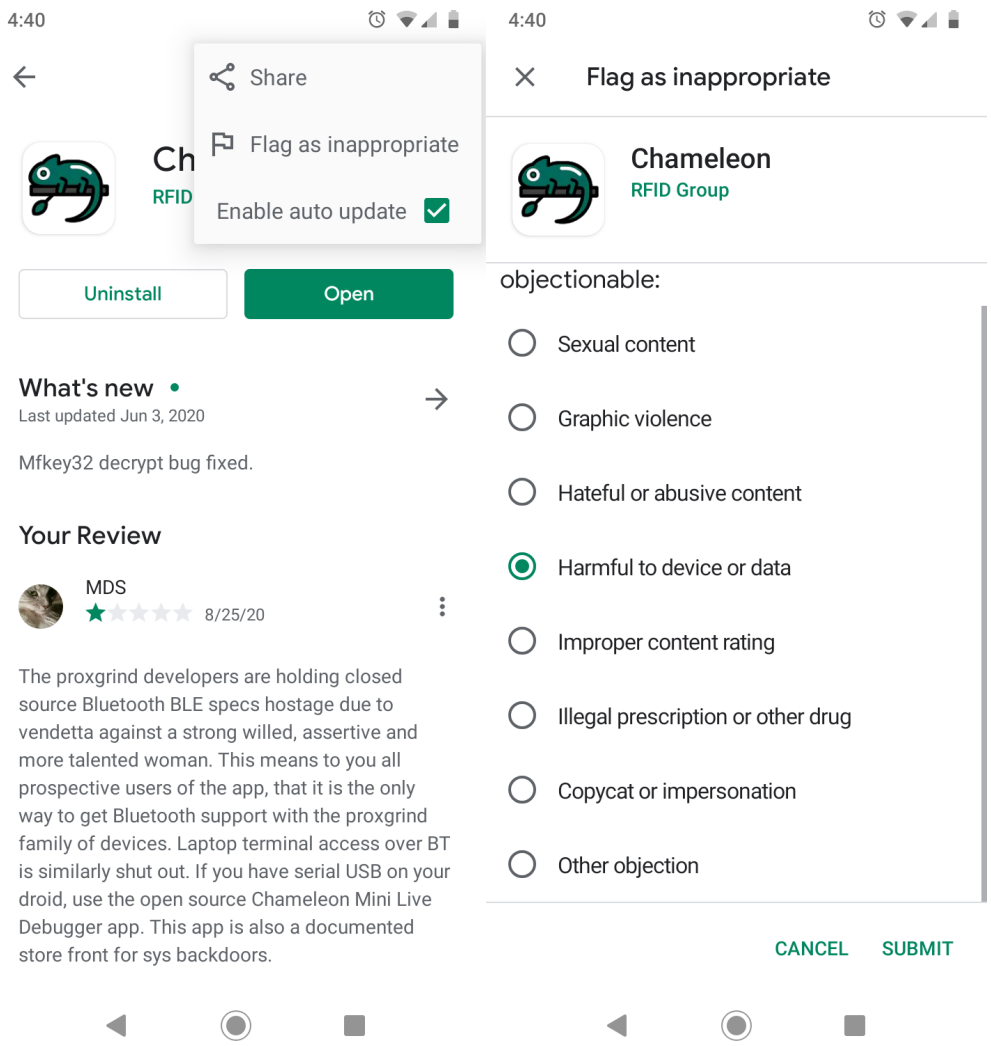
maxieds commented on Aug 24, 2020 • edited Contributor Author

@david-oswald
I am going to go farther than what you did to insinuate that this app loads malware. Based on my public inspection of the obfuscated reversed engineered APK sources to the `com.proxgrind.chameleon` app (see below), it is evident to me that the following is correct: **The vendor app is nothing more than a store front installer for more backdoor hooks into the system to come. Not unlike some of the services Facebook puts on a droid with their characteristically interpreted, but more commonplace, spyware application.**

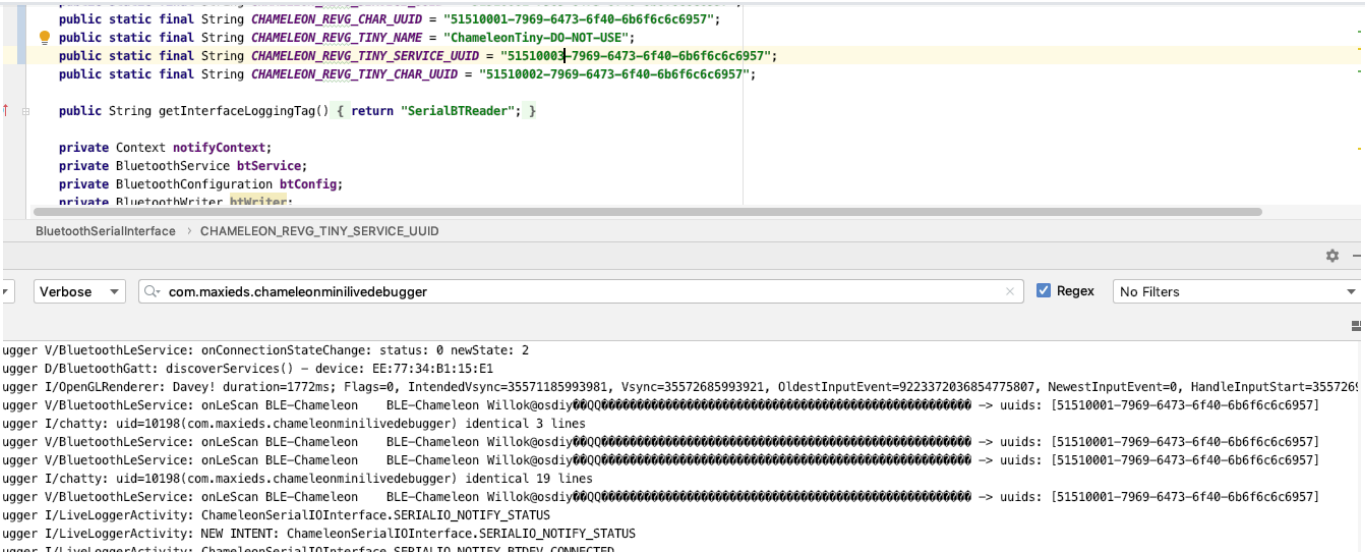
If I were a concerned user, or otherwise security conscious person, I would take the following course of action about having unwittingly been duped into installing this app on my Android device under the premise that it is legitimate, good natured vendor code to control my new (pricey) Chameleon device with Bluetooth:

1. Uninstall the application

3. Perform a factory reset of the device to remove all backdoor malware and suspicious hidden activities it could have put on your system
4. Let the Proxgrind developers and **the merchant your purchased the Proxgrind device from** know you are unhappy



In principle, users that were forced into using the vendor application to get otherwise obscured, and incomplete, BLE Bluetooth connections to their Proxgrind Chameleon can expect that ability from my app some time in the future. With respect to BT not working in the *CMLD* app (again, my original GPL, freely available open source Android alternative to the vendor), it is not going to be readily possible without more specific documentation about the BT BLE device from the vendor. I have inquired about getting these details, but they have chosen on their own volition to blow me off. This could be an artifact of my choice of 4-letter language at them when I discovered this malware feature set in their closed-source Droid app. If you want them to communicate more about these specs to me to get BT working in the *CMLD*, please do drop them a line and let them (and their CTO) know you are unhappy.



It also makes me nervous that unlike the AVR firmware sources, the OTG Bluetooth firmware is conspicuously closed-source and tenuously available to users only in binary form. Again, if that makes you nervous, like it does to me, please drop the Proxgrind vendors an issue and let them know you are unhappy about it.

Telltale signs that the com.proxgrind.chameleon Android application is doing malicious things

Explanation of dynamic code loading (possibilities versus reality)

It is possible based on my reading of the reversed engineered Java code for the app that it uses a dynamic loader to import functionality from pre-compiled *.so* (shared object) files, presumably obtained from C source, under a number of platform-specific variants (e.g., processor types). This could be explained by the app's builtin mfoc (Mifare offline cracker) functionality. In other words, it is possible that whomever developed this Android app was either inexperienced, or just plain lazy, and did not realize that the C/C++ sources for that code can be directly imported and included in native code mode. However, the presence of other shared object libraries included with the APK binary is concerning. In general, this sort of procedure on Android is bad practice, on the order of at least a code smell, and should be discouraged, possibly with the exception of those who would have signed keys and can write system apps -- but even then, I don't like this layer of obfuscation. I continue below with other symptoms of malware functionality.

No a priori semblance of Android BT functionality in the app before dynamically loading remote object from com.qihoo.util

evidence of past bad behavior) is very disconcerting given the bad reputation and track record from that source.

Presence of troublesome on-boot (and DFU) services and options in the manifest file

See the description of DFU functionality and the following snippet taken from the `com.proxgrind.chameleon` Android manifest file uploaded here:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.proxgrind.c
    <uses-sdk android:minSdkVersion="19" android:targetSdkVersion="29"/>
    <uses-permission android:name="android.permission.NFC"/>
    <uses-permission android:name="android.permission.VIBRATE"/>
    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission-sdk-23 android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <application android:theme="@style/AppTheme" android:label="@string/app_name" android:ico
        <service android:name="com.proxgrind.chameleon.services.RestartService" android:enabl
        <service android:name="com.proxgrind.chameleon.services.DfuService">
            <intent-filter>
                <action android:name="no.nordicsemi.android.action.DFU_UPLOAD"/>
                <category android:name="android.intent.category.DEFAULT"/>
            </intent-filter>
        </service>
        <service android:name="com.proxgrind.chameleon.services.BondService" android:exported
```

👀 1 ☺

maxieds commented on Aug 24, 2020 • edited Contributor Author

On second read of the third point above, it is possible that the DFU service is how they flash the BT firmware? But, WTF is the restart --**service**-- about? I am very concerned that this app has the capability to turn my Droid into a remote drone.

Let me also reiterate the insensitive manner in which I was responded to by the Proxgrind developer contact point (as of yet, not word, nor input from a very prominent merchant in the EU about their status on this, at least to me):

> Re: Concerns with the Proxgrind Android app (com.proxgrind.chameleon)

> more details on your concerns and how we can achieve more together ?

Update:

"My problem" seems to be an innate concern for users in the Chameleon Mini community, to which I have been readily forthcoming and able to contribute my software piece, for years. How absolutely irresponsible and reckless would I need to be to stay silent about this?

 **maxieds** commented on Aug 24, 2020 Contributor Author

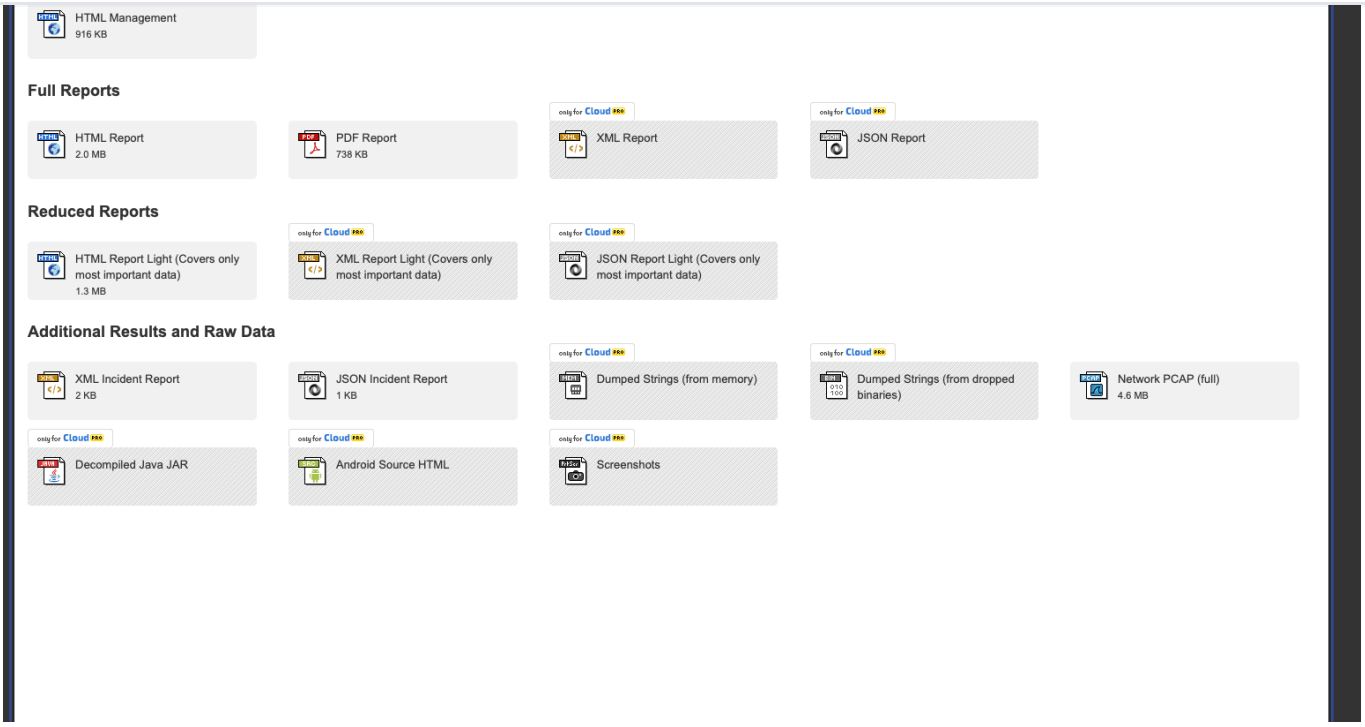
At someone's suggestion today, I am trying to be more polite to strangers. Some strangers deserve it, others clearly do not. I am therefore appealing to a subset of the group of smart masses here with raw data. My opinion about this app being malicious is stated above. Please do feel free to draw your own educated opinions based on the following data:

Preliminary reconnaissance: Google search provides links

I have a few other sparse links searching around for more information:

- Android: Create a stub authenticator: *The sync adapter framework assumes that your sync adapter transfers data between device storage associated with an account and server storage that requires login access.*
- NXP: com.qihoo.util reference
- Apparent StubApp hook: Very concerning, be careful about clicking around this link
- com.qihoo.security APK analysis
- A good analysis result for Android APKs

Anyone want to sign up for a Pro account to get dumped memory strings?



Interesting IP traffic from the default com.proxgrind.chameleon APK install

AV Detection:

Multi AV Scanner detection for submitted file

Source: com.proxgrind.chameleon_2020-06-03_1803.apk	Virustotal: Detection: 9%
Source: global traffic	TCP traffic: 192.168.2.30:55938 -> 8.8.4.4:853
Source: global traffic	TCP traffic: 192.168.2.30:44524 -> 8.8.8.8:853
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.21.14
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46

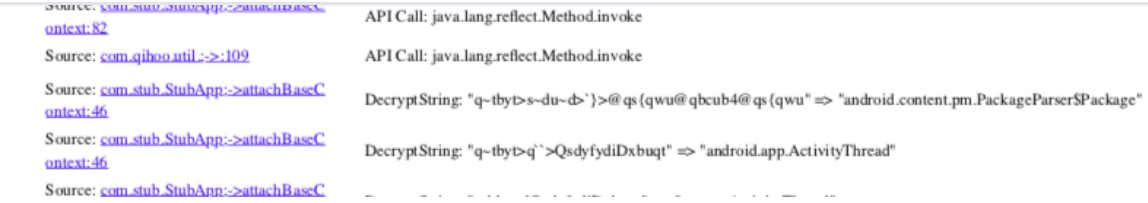
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46
Source: unknown	TCP traffic detected without corresponding DNS query: 172.217.19.46

https://www.joesandbox.com/analysis/276338/0/executive

Source: libmfkey64.so, libcraptol.so, libnested.so	String found in binary or memory: https://android.googlesource.com/toolchain/clang
Source: libmfkey64.so, libcraptol.so, libnested.so	String found in binary or memory: https://android.googlesource.com/toolchain/llvm
Source: unknown	Network traffic detected: HTTP traffic on port 36510 -> 443
Source: unknown	Network traffic detected: HTTP traffic on port 443 -> 38372
Source: unknown	Network traffic detected: HTTP traffic on port 43534 -> 443
Source: unknown	Network traffic detected: HTTP traffic on port 443 -> 36510
Source: unknown	Network traffic detected: HTTP traffic on port 44226 -> 443
Source: unknown	Network traffic detected: HTTP traffic on port 38372 -> 443
Source: unknown	Network traffic detected: HTTP traffic on port 47426 -> 443
Source: unknown	Network traffic detected: HTTP traffic on port 443 -> 43534
Source: unknown	Network traffic detected: HTTP traffic on port 443 -> 47426
Source: unknown	Network traffic detected: HTTP traffic on port 443 -> 44226
Source: submitted apk	Request permission: android.permission.BLUETOOTH
Source: submitted apk	Request permission: android.permission.BLUETOOTH_ADMIN
Source: submitted apk	Request permission: android.permission.WRITE_EXTERNAL_STORAGE
Source: classification engine	Classification label: mal48.andAPK@0/251@1/0
Source: com.qihoo.util.DtcLoader:-><clinit>:2	API Call: java.lang.System.loadLibrary ("jgdtc")
Source: com.stub.StubApp:->attachBaseContext:69	API Call: java.lang.System.loadLibrary ("X86Bridge")

https://www.joesandbox.com/analysis/276338/0/executive

Source: com.stub.StubApp:->attachBaseContext:72	API Call: java.lang.System.loadLibrary ("jiagu_x86")
Source: com.stub.StubApp:->attachBaseContext:86	API Call: java.lang.System.loadLibrary ("jiagu")
Source: com.proxgrind.chameleon_2020-06-03_1803.apk	Total valid method names: 50%



Relevant PDF data (please read through 1 and 3!)

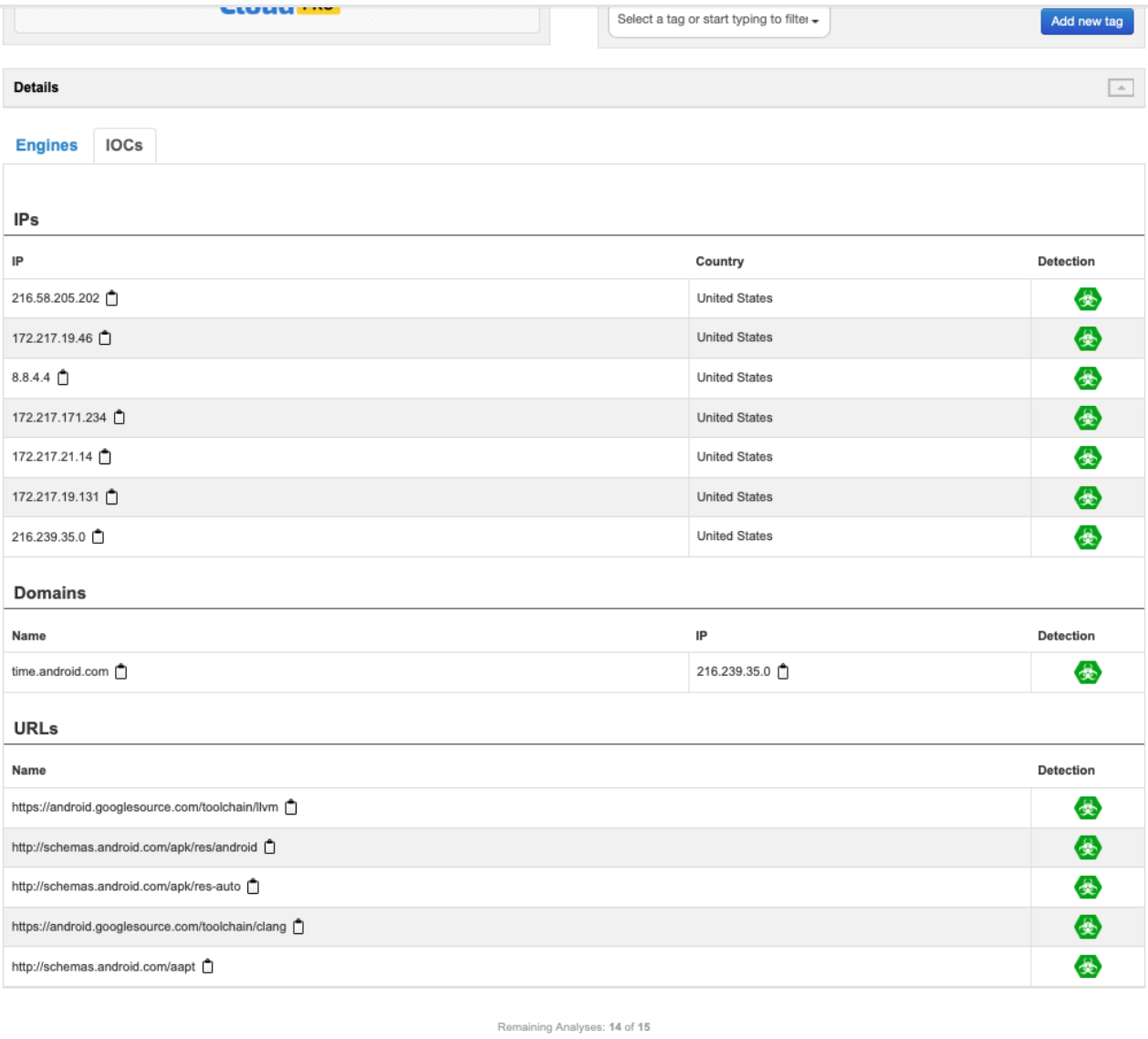
AndroidManifest.xml.pdf
APKStaticAnalysis.pdf
report-a8dc7c27a273a56dbc084bca7dcf03f5.pdf

🚀 1 👀 1 ☺

maxieds commented on Aug 24, 2020 • edited Contributor Author

It also occurs to me to provide the dumped network traffic PCAP file downloaded from the analysis here. Enjoy!

dump-a8dc7c27a273a56dbc084bca7dcf03f5.pcap.txt



maxieds commented on Aug 24, 2020 Contributor Author

For comparison, I am also remitting the analogous analysis for the previous (*com_proxgrind_chameleon_2020-02-18_1547_02_17_2020.apk*) APK, and for the record and comparison, for my app, the Chameleon Mini Live Debugger (my own local signed developer copy: *app-free-release.apk*):

report-1ae2b86c9ca728aee93fc699e3cb04b2.pdf -- Analysis of the older version of the com.proxgrind.chameleon APK -- **MORE IPs LISTED**

report-270665c7f6399bf6c4119808584eb0c5.pdf -- CMLD application: Verdict, CLEAN application!

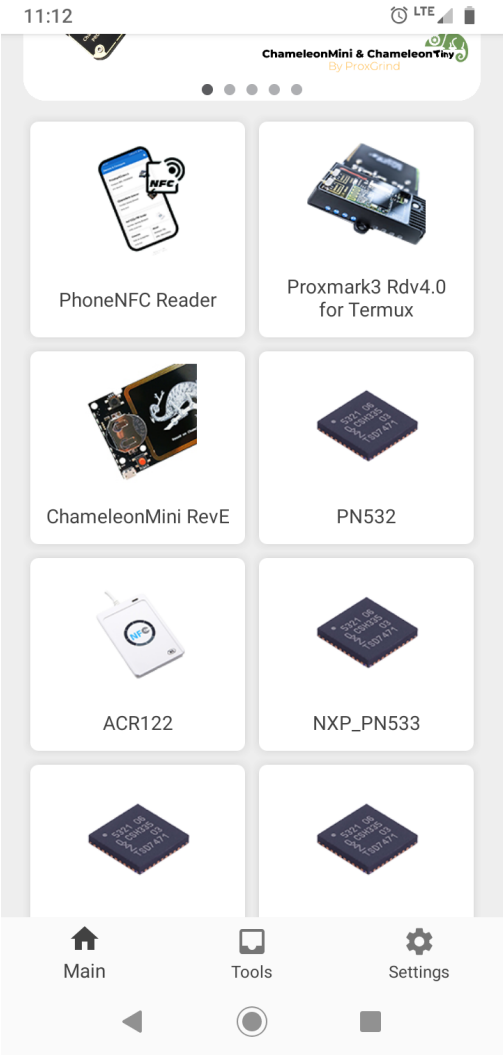
maxieds commented on Aug 24, 2020 • edited Contributor Author

@david-oswald: I lack a discord invite.

Unfortunately, I still have pertinent information that I feel is risky not to make available to users. I am particularly concerned given the following `res/xml/arrays.xml` entry:

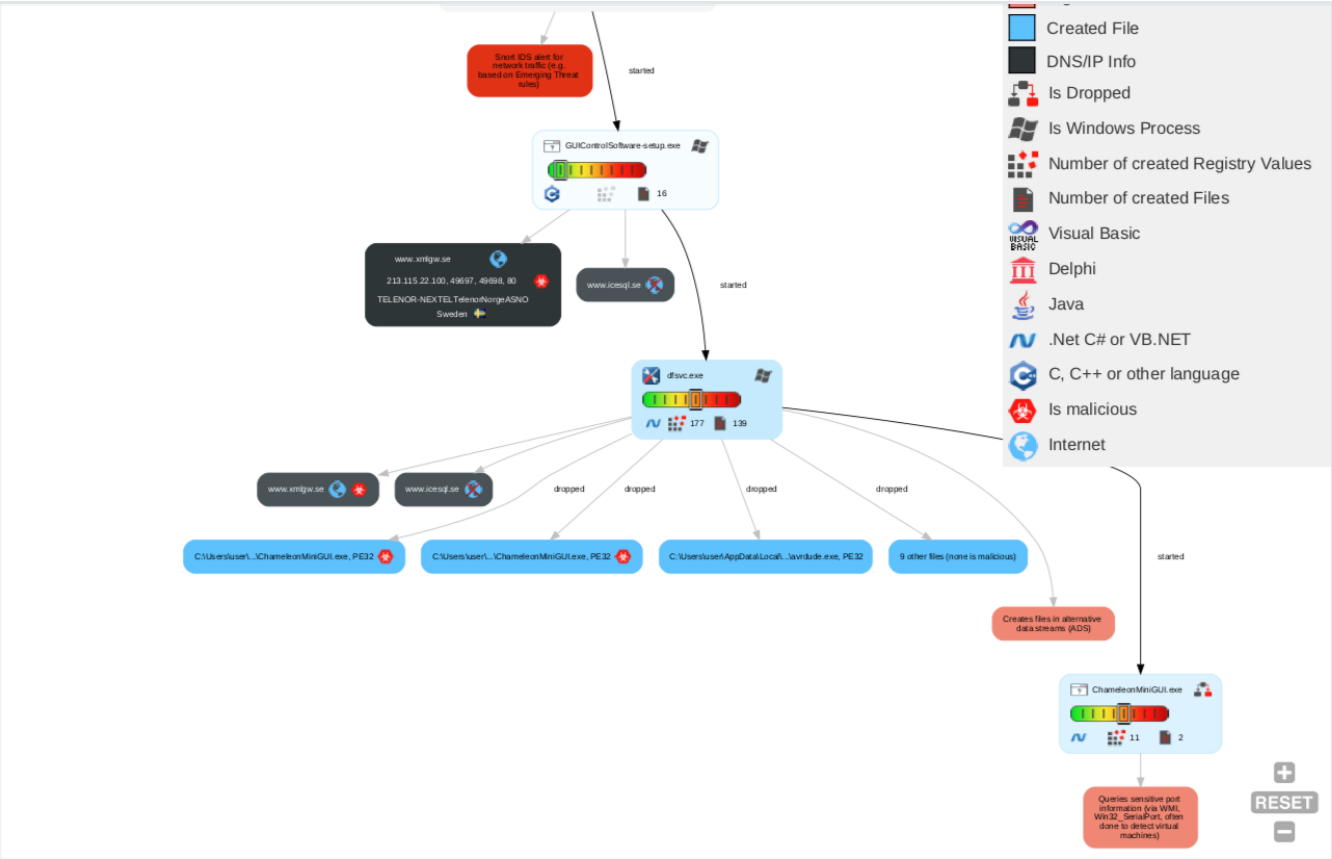
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <array name="chameleon_click_list">
        <item>换卡</item>
        <item>随机卡号</item>
        <item>卡号左增</item>
        <item>卡号右增</item>
        <item>卡号左减</item>
        <item>卡号右减</item>
        <item>关闭动作</item>
    </array>
    <array name="chameleon_size_list">
        <item>@null</item>
        <item>64</item>
        <item>1024</item>
        <item>4096</item>
    </array>
    <array name="chameleon_slot_list">
        <item>卡一</item>
        <item>卡二</item>
        <item>卡三</item>
        <item>卡四</item>
        <item>卡五</item>
        <item>卡六</item>
        <item>卡七</item>
        <item>卡八</item>
    </array>
    <array name="chameleon_type_list">
        <item>关闭</item>
        <item>UL</item>
        <item>1K</item>
        <item>4K</item>
        <item>侦测</item>
    </array>
    <array name="device_list_support">
        <item>1、自带</item>
        <item>2、PN532_SPP</item>
        <item>3、PN532_OTG</item>
        <item>4、Acr122u_OTG</item>
        <item>5、Proxmark3_OTG</item>
        <item>6、Chameleon_OTG</item>
```

In particular, this looks much like the profile from the RFIDTools Android app, a common controller for the Proxgrind3 and other NFC related devices (see screenshot below):



~~I am also concerned that if this same vendor is involved in making the next generation of NFC tools (e.g., proxmark devices), then this disturbing trend can easily persist.~~ (**UPDATE:** no problems here with that app)

Other software available is concerning:



AutomatedMalwareAnalysisExecutiveReport.pdf
dump-9dac98579289a7e0ffbad3fef093f279.pcap.txt

Update:

N.b., I am not a security professional.

maxieds added a commit to maxieds/ChameleonProxgrindAndroid-FullDisclosure that referenced this issue on Aug 25, 2020

`Preserving more binary data` ··· `ea2c08e`

aveao commented on Aug 25, 2020 • edited

Contributor

Disclaimer, note, whatever: I am **not** related to RRG, proxgrind, et al. I just want to ensure that the product I invested in is safe, just like you appear to be.

I'll be writing my stuff I've discovered over at Discord here too.

Firstly, regarding your post.

Regarding qihoo, I've found references to qihoo offering packing solutions on page 8 of https://arxiv.org/pdf/1801.01633.pdf, as this product: https://jiagu.360.cn/

While this is obviously undesirable, that alone doesn't jump directly to "malware" for me.

Regarding USB devices,

They are the following, in this specific order:

- Future Technology Devices International, Ltd - FT232 Serial (UART) IC - 0403:6001
- Future Technology Devices International, Ltd - Bridge(I2C/SPI/UART/FIFO) - 0403:6005
- Arduino SA - All - 2341:All
- Van Ooijen Technische Informatica - Teensyduino Serial - 16c0:0483
- Cygnal Integrated Products, Inc. - CP2102/CP2109 UART Bridge Controller [CP210x family] - 10c4:ea60
- Prolific Technology, Inc. - PL2303 Serial Port - 067b:2303
- QinHeng Electronics - HL-340 USB-Serial adapter - 1a86:7523

They all appear to be, well, UART and serial connectors. ChameleonMini/Tiny also works over USB and uses serial.

Several of these appear to be relics from development, and I assume only two or so of these are currently in use by production devices, but none of this appears malicious to me.

> Explanation of dynamic code loading (possibilities versus reality)

> No a priori semblance of Android BT functionality in the app before dynamically loading remote object from com.qihoo.util

> Preliminary reconnaissance: Google search provides links

appears to be the same as qihoo point above.

> Presence of troublesome on-boot (and DFU) services and options in the manifest file

I'm not sure what this is, though it might be related to updates. Restart might be about restarting the chameleon after an update, though I think dennis can say more about that.

> Anyone want to sign up for a Pro account to get dumped memory strings?

> Interesting IP traffic from the default com.proxgrind.chameleon APK install

Those are Google IPs, so I assume they're just pinging google to verify that network is connected, that'd also explain the DNS query.

maxieds commented on Aug 25, 2020 • edited Contributor Author

@aveao
What alarms me the most out of the reactions I'm getting is that no one seems outraged! Throwing 💩 against the wall to see what sticks, or otherwise obfuscates things beyond all recognition, is not going to fly with me on this day. I do however respect you up front for debating this with me in public. This is not about you personally. The insinuation that burden of malice is on me is not a good attitude, and moreover, given the evidence I have documented above it has already been done.

To paraphrase Marvel comics,

Let me respond to what you just wrote again in slightly different wording:

What I am now proposing after some reading is that parts of the Android source for the com.proxgrind.chameleon application are based (loosely) on the open source RFIDTools Droid app. This explains the UART devices you mentioned found in the specs for the `/res/raw/device_filter.xml` above. It also explains to some extent the entries in `/res/xml/arrays.xml` referenced above here.

one option. There are other techniques under which these ports get used to bypass otherwise stringent firewall rulesets. Why is there network traffic being initiated by the app anyway? If it's just a loader for *.so* binaries to protect proprietary investments in code (as suggested by your arXiv link), then everything they need to load is already in the APK's included assets folder, right? This all smells very bad to me.

What is most disconcerting to me is the Bluetooth issues linked here:

1. The BT firmware for the Proxgrind devices is closed-source. Why? Also, it is either broken, or a very proprietary build of a BLE device spec that doesn't play nicely with others. Documentation does not exist and is not forthcoming from the Proxgrind developer people.
2. When you look at the JADX reverse engineered APK sources (in Java, Kotlin, shared object binary form) from here, there is **ABSOLUTELY NO** Bluetooth functionality from the standard Android BT stack. One then gets very suspicious what put together this all means?

Re: packing solutions on page 8 of https://arxiv.org/pdf/1801.01633.pdf:

What exactly in the Proxgrind app needs to be protected in this way? Things like the `mfcrack32.so` are (should be) based on the open source MFOC tool written in C/C++. What is being hidden here? What I'm saying, having done a lot of Android hacking myself, is that it is so abnormal to have no traces of the Android BT libraries controlling the BT functionality in the app, that it is almost impossible that this StubApp is doing legit things.

Compare this output with the output of this tool on the RFIDTools APK (off Play Store). Night and day really. One wreaks of malware, the other is a typically packaged Droid app, for example. They both have pre-compiled binary *.so* assets! Similar technique. The difference is one is completely lacking the Android backing declared in the `AndroidManifest.xml`. See also this link for a general evasive procedure.

There is also again a point to make about the issue with web traffic initiated by the app. See the sandbox sites output. Things are not as innocent as they appear. Run it for yourself with a suspicious eye considering what I have written and seen so far above!

maxieds commented on Aug 25, 2020 Contributor Author

Another very conspicuous sign of foul play to me is evidenced by a subtle declaration of the permissions in *com.proxgrind.chameleon*:

- android.permission.BLUETOOTH_ADMIN
- android.permission.FOREGROUND_SERVICE
- android.permission.NFC
- android.permission.READ_EXTERNAL_STORAGE
- android.permission.VIBRATE
- android.permission.WRITE_EXTERNAL_STORAGE

As specified in the Android BT docs, the application either needs to declare a location permission like `ACCESS_COARSE_LOCATION` (missing in the vendor app), or otherwise use the CompanionDeviceManager API to handle BT pairing. However, as noted in the docs, to use this secondary method, we require the following **missing** addition to the `AndroidManifest.xml` for the app:

```
<uses-feature android:name="android.software.companion_device_setup"/>
```

It stands to reason that this is a clear demonstration that the BT functionality in *com.proxgrind.chameleon* does not go through the Android APIs. Another highly suspect insight into the vendor app not behaving itself well on my system (anthropomorphism implied). **QED**

Akisame-AI mentioned this issue on Aug 25, 2020

Adding support to the firmware for DESFire emulation? #218

Closed

maxieds mentioned this issue on Aug 25, 2020

Concerned with tampered APK sources on Play Store (1.1.6-free)

maxieds/ChameleonMiniLiveDebugger#29

Closed

david-oswald commented on Sep 9, 2020 Collaborator

I'll close this issue now, as it doesn't really relate to the Chameleon development in this repo.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

None yet

3 participants