

# Anzu Raptor Drone, RC and App

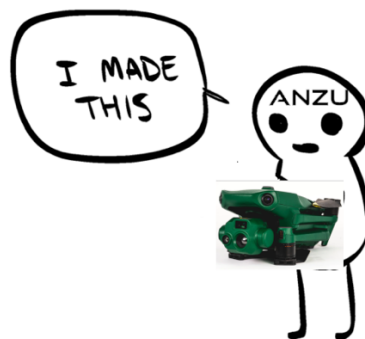
## A quick analysis

Author:

Andreas Makris aka Bin4ry  
[am@think-awesome.com]

Co-Authors:

Kevin Finisterre aka d0tslash  
Jon Sawyer aka jcase



May 18<sup>th</sup> 2024

Thanks to the two anonymous folks that independently contacted about the whole "white label" thing

Anzu Robotics (<https://www.Anzurobotics.com/>) is an US based company that got some attention in the last weeks by releasing an DJI drone lookalike. The drone they release is called "Raptor (T)", it shares the Specs with the DJI Mavic 3 Enterprise / Mavic 3 Enterprise Thermal. Anzu itself has put an FAQ on their website explaining that they have licensed the technology from DJI.

#### ^ WHAT IS THE ONGOING INVOLVEMENT OF THE LICENSING ORGANIZATION (DJI)?

Licensing technology is a normal business practice. Our agreement in licensing the technology associated with the Raptor-series drones gave Anzu Robotics the rights to modify and manufacture this technology at will. There are no royalties shared with the licensing organization, no joint or shared ownership of Anzu Robotics, and no reporting on customer data. There may be additional products developed through similar agreements, but our work at Anzu Robotics is independent of any other party aside from the transfer of this technology.

Of course, licensing technology is a standard business practice, but with the foreshadowing of an ban of DJI products in the United States the big question is: How much DJI is in the Anzu device, or better said, is the Anzu device really an own device or just an green painted Mavic 3?

Anzu products are produced in Malaysia (maybe the DJI Factory? ;))

#### ^ WHERE ARE ANZU ROBOTICS PRODUCTS MANUFACTURED?

Nearly all of the components and final assembly of our drones is done in Malaysia. The completed drone hardware is sent to Anzu Robotics in the United States and firmware is then installed and quality control measures are conducted.

As the FCC filings (<https://fcc.report/company/Anzu-Robotics-L-L-C>) have shown the products are identical on the hardware level to the DJI products. The Anzu drones use the DJI proprietary hardware such as the P1 (Pigeon) etc. Our friends of the FPV wiki have already put the PCBs side by side: <https://fpvwiki.co.uk/dji-white-label-clone-drones> and came to the conclusion that they are identical.

Having in mind that the hardware is the same and that we have to determine if the Anzu drone is its own device or just a green DJI Mavic 3 Enterprise we need to take a look at the drone firmware. Let's see what Anzu tells us about the firmware in their FAQ.

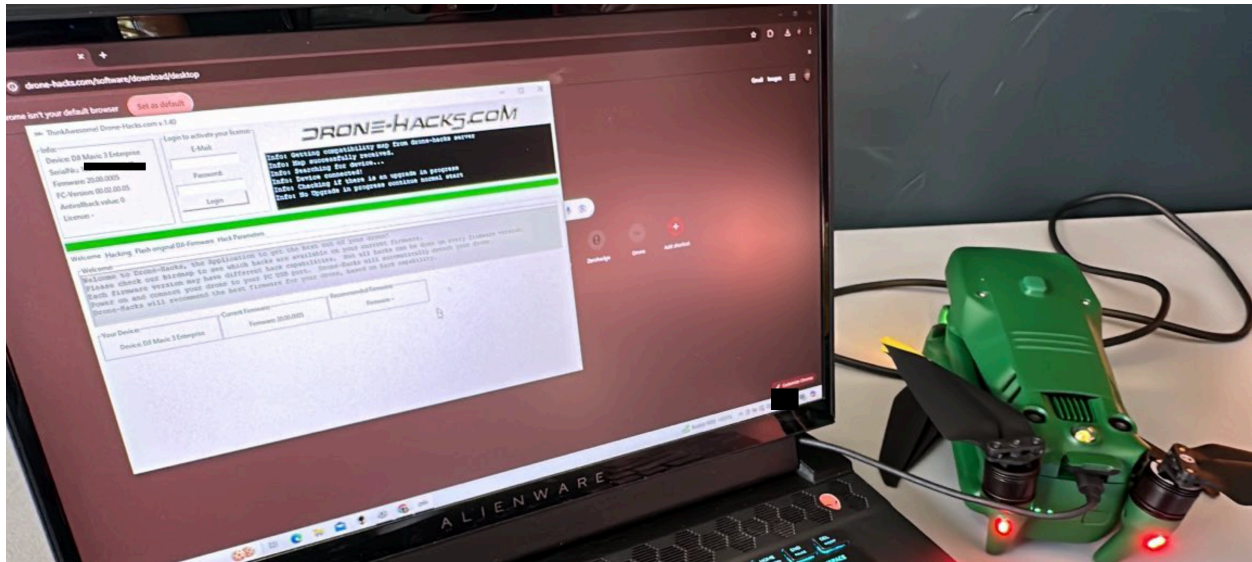
## ^ WHERE DOES THE FIRMWARE ON THE RAPTOR-SERIES DRONES COME FROM?

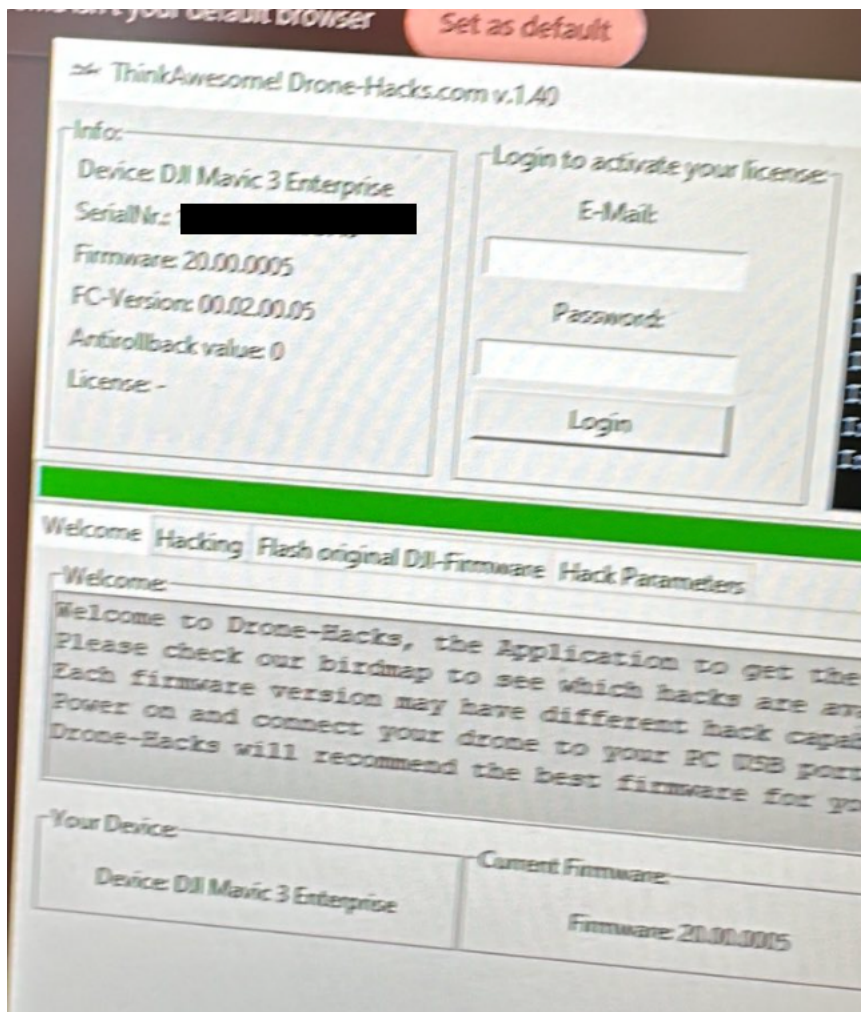
When we finalized the licensing agreement we were provided a version of custom firmware as a snapshot in time. That firmware was then housed on Aloft's servers for development and can't be accessed or modified by anyone else. The flight dynamics of our drones were not redeveloped, so you're getting world-class performance benefitting from more than a decade of aerial robotics R&D. We have taken the necessary measures to ensure that anything on our drones is void of data security vulnerabilities.

When Anzu writes something like this it implies that there was a custom build for the hardware done and it does not run any DJI software anymore. Of course, they do not write that explicit, but calling it an custom firmware implies that there is something special to the firmware, except the flight dynamics, which were not redeveloped. As far as I am concerned, I would expect everything else to be redeveloped when I read this sentence. Furthermore the firmware is hold at Aloft, not DJI. Great! Let's take a look.

What we did is to get the raptor drone and connect it to our computer. The very first thing we checked is if the drone reacts to normal DJI commands and can be detected by software designed unique for DJI drones. This software, the drone-hacks.com software, speaks to the drone via DJI's DUMI protocol and is able to read/flash/dump several information from the drone when connected.

This is what the drone-hacks.com software showed when connecting the Raptor drone:





The drone-hacks.com software detects the drone as a Mavic 3 Enterprise, which is interesting. If this is a custom firmware, shouldn't it be at least renamed? Well let's see what the firmware version's cfg file shows:

```

<dji>
  <device id="m265e">
    <firmware format="20.00.0005">
      <release version="20.00.0005" antirollback="0" antirollback_ext="cn:0" enforce="0" enforce_ext="cn:0" enforce_time="2023-12-12T16:11:37+00:00" from="2023/12/13" expire="2024/12/13">
        <module id="0802" version="20.00.00.07" type="" group="ac" order="5" wait="3" size="221224960" name="WM265E_E2" upgrade_order="5" is_upgrade_center="true" op_lib_name="libeagle_m" />
        <module id="1502" version="20.00.00.04" type="" group="ac" order="4" wait="2" size="216776640" name="WM265_E1E" upgrade_order="4" is_upgrade_center="false" op_lib_name="libstanda" />
        <module id="1100" version="08.75.02.23" type="" group="ac" order="1" wait="1" size="96500" name="Battery" upgrade_order="1" is_upgrade_center="false" op_lib_name="libstandar_md" />
        <module id="1200" version="01.90.01.27" type="mc01" group="ac" order="3" wait="0" size="118496" name="ESC_0" upgrade_order="3" is_upgrade_center="false" op_lib_name="libstandar_md" />
        <module id="1202" version="01.90.01.27" type="mc01" group="ac" order="2" wait="0" size="118496" name="ESC_2" upgrade_order="2" is_upgrade_center="false" op_lib_name="libstandar_md" />
        <module id="0105" version="01.13.10.01" type="" group="ac" order="4" wait="0" size="285472" name="LCPU" upgrade_order="4" is_upgrade_center="false" op_lib_name="libstandar_md_up" />
        <module id="0500" version="05.02.22.46" type="" group="ac" order="3" wait="0" size="91200" name="PD_MCU" upgrade_order="3" is_upgrade_center="false" op_lib_name="libstandar_md_u" />
        <module id="1006" version="01.00.00.28" type="pa01" group="ac" order="3" wait="0" size="253184" name="SPEAKER_MCU" upgrade_order="3" is_upgrade_center="false" op_lib_name="libsta" />
        <module id="2607" version="00.00.79.80" type="" group="ac" order="3" wait="0" size="2435936" name="RTK982" upgrade_order="3" is_upgrade_center="false" op_lib_name="libstandar_md" />
      </release>
    </firmware>
  </device>
</dji>

```

The cfg file is a signed configuration file of the current drone firmware. The drone holds a copy of this file on its own filesystem, the file is used during the flashing process of the firmware to ensure only firmware can be flashed which is signed cryptographically with the correct key.

The cfg file itself is also signed, so it cannot be altered without being voided. The cfg file references several modules and holds their checksums.

```
ll" md5_unsign="07a874abe301f5f9fd7ad1969beb5f3c" md5="1df8b50a0de0f0cdc1a37e1120a03bcd">wm265e_0802_v20.00.00.07_20231212.pro.fw.sig</module>
_tota="null" md5_unsign="232fcc8de2001c0423edabaf6796baf" md5="d70eb1fe7b428e492329d848c5a6c648">wm265e_1502_v20.00.00.04_20231212.pro.fw.sig</module>
" md5_unsign="04c152b91590b37012afa1aa81ff4367" md5="961de38a71185b29e925d854751b0036">wm265e_1100_v08.75.02.23_20230731.pro.fw.sig</module>
null" md5_unsign="11a6957532b25a65a722302a2ada34" md5="436cbf194eff0e243e0b82727e8643bc">wm265e_1200_v01.90.01.27_20221208_mc01.pro.fw.sig</module>
null" md5_unsign="a0e2f8205a4e0c9d3a0b754ac1c36f49" md5="bcad2e24f2b9d144f10ff2fb91c6cdc5">wm265e_1202_v01.90.01.27_20221208_mc01.pro.fw.sig</module>
md5_unsign="31ef8d3c60ef0fa8fb9252b7f672a3d1" md5="fb04a6ff83f8689965f9498065d7aa7e">wm265e_0105_v01.13.10.01_20230511.pro.fw.sig</module>
" md5_unsign="37cb7bce7793220445684645fd21a86a" md5="08e5c3cb71d70e009a072c902f435c29">wm265e_0500_v05.02.22.46_20210825.pro.fw.sig</module>
a="null" md5_unsign="d35738021208a1a5987a0d4c14e04d72" md5="57b13ae9a32def57f1dc76180dd0c26e">wm265e_1006_v01.00.00.28_20221123_pa01.pro.fw.sig</module>
l" md5_unsign="4bdb604fdb4316843f8e9d232c150643" md5="0bbebc88b4acaa4df7c1b561c40fc49a">wm265e_2607_v00.00.79.80_20221201.pro.fw.sig</module>
```

During the flashing process the cfg file is parsed and every module is checked for its checksum, only if the checksum is correct for the module said module will be processed. Every module itself is then checked for the signature and only if valid it will be decrypted and flashed.

The keys used for to verify the signature and decrypt the firmware are handled by the trustzone.

Ultimately if Aloft is in control of the firmware, they should of course use their own keys and not rely on DJI keys, if they would rely on DJI keys they would not be in control of the firmware, as they would not be able to encrypt and sign the images themselves but would have to send them to China for processing.

For the Mavic 3 Enterprise I happen to be able to decrypt the firmware images with the correct key, if the Raptor firmware is signed with an own key the Mavic 3 Enterprise key should not work. Let's see:

```
(env) → wm265e md5 wm265e_0802_v20.00.00.07_20231212.pro.fw.sig
MD5 (wm265e_0802_v20.00.00.07_20231212.pro.fw.sig) = 1df8b50a0de0f0cdc1a37e1120a03bcd
(env) → wm265e python3 decrypt.py wm265e_0802_v20.00.00.07_20231212.pro.fw.sig
Mavic 3 Enterprise key successful
```

As you can see the md5 of the 0802 module image is the same as from the cfg file, therefore we verified that we are handling the correct firmware image file here. Then the decrypter script found that the Mavic 3 Enterprise key was successful, it was able to decrypt the file using the Mavic 3 Enterprise key, interesting...

Let's check if the md5 of the decrypted file is the same as the "md5\_unsign" md5 from the cfg file:

```
(env) → wm265e md5 wm265e_0802_v20.00.00.07_20231212.pro.fw.sig.0802.dec
MD5 (wm265e_0802_v20.00.00.07_20231212.pro.fw.sig.0802.dec) = 07a874abe301f5f9fd7ad1969beb5f3c
```

That looks correct, so I have indeed been able to verify the decryption of the file.

What does that mean? The firmware was being signed and encrypted using the very same key as the normal Mavic 3 Enterprise, it is not signed and encrypted by Aloft or Anzu themselves, but by DJI. The claim that nobody else can modify the firmware images of the drone is therefore false, not even Aloft can modify it without voiding it, only DJI can.

Additionally, Anzu claims that the firmware is hold on the Aloft servers and can't be accessed by anyone else.


Let's check: I was able to check the DJI server for the firmware, same as DJI Assistant 2 tool one can manually download firmware for updates. Let's see if we download the 0802 firmware from DJI directly, click this link and see what you get:

[https://terra-2-g.djicdn.com/1546577435024199aad91154298e6ca7/apddnhlhvae2v6aipfgr2v2q/wm265e\\_0802\\_v20.00.00.07\\_20231212.pro.fw.sig?auth\\_key=1716088014-1716080814670-0-e0c729a9921498f9493b314134252b1b](https://terra-2-g.djicdn.com/1546577435024199aad91154298e6ca7/apddnhlhvae2v6aipfgr2v2q/wm265e_0802_v20.00.00.07_20231212.pro.fw.sig?auth_key=1716088014-1716080814670-0-e0c729a9921498f9493b314134252b1b)

Correct: There is our Anzu firmware living on the DJI CDN, so the firmware is also hosted by DJI in contrary to the claims from Anzu.

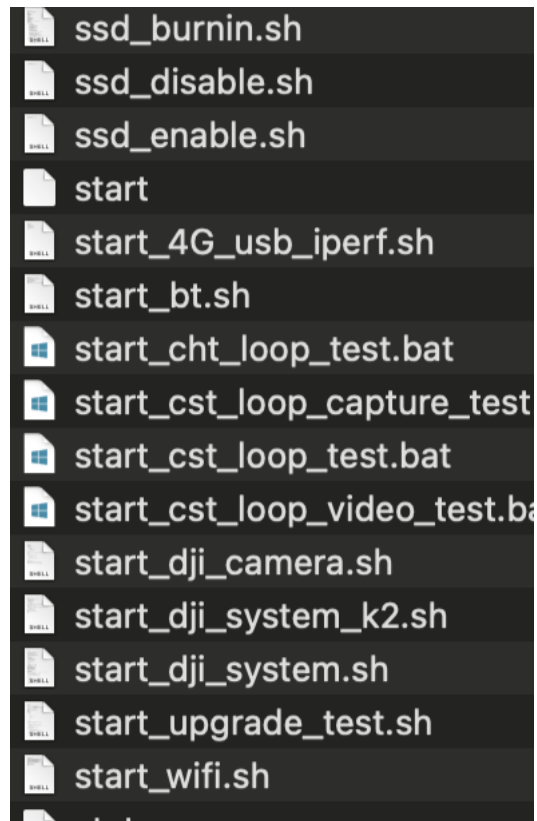
Let's look what is going on in the drone firmware itself. I will check the 0802 module as an example again, if this is a custom firmware I would expect Aloft/Anzu software and not DJI software in this module. As Anzu claimed in the FAQ the software is custom, except the flight dynamics, so I would expect everything but the flightcontroller (which is not part of the 0802 module!) to be custom.

This is part of the system/bin folder:



- dji\_amt
- dji\_blackbox
- dji\_camera3
- dji\_cht
- dji\_config\_net\_route.s
- dji\_config\_store
- dji\_cpu\_pll\_frequency
- dji\_crashdump.sh
- dji\_cst
- dji\_dcs
- dji\_dsp\_load
- dji\_fcali\_test
- dji\_fnm\_test
- dji\_ftpd
- dji\_fulldump
- dji\_fw\_load
- dji\_fw\_verify
- dji\_gcov
- dji\_hms
- dji\_kmsg
- dji\_lte
- dji\_mb\_ctrl
- dji\_mb\_ctrl\_safe.sh
- dji\_mb\_parser
- dji\_media\_play\_servic
- dji\_media\_player\_test
- dji\_ml
- dji\_ml\_gtest
- dji\_nn\_server
- dji\_perception
- dji\_perf.sh
- dji\_pinmux\_check
- dji\_ppt
- dji\_production\_check
- dji\_sdrs\_agent
- dji\_sec
- dji\_sn\_ops.sh
- dji\_sw\_uav
- dji\_sys
- dji\_tombstone.sh
- dji\_top
- dji\_upgrade
- dji\_vtwo\_sdk
- dji\_wlm\_slave
- dji\_wm265e\_ca...ra\_a
- dji\_wm265e\_ca...re\_a
- dmesg

Pretty much standard DJI binaries are there, this is the exact layout as the normal Mavic 3 Enterprise, nothing special here, nothing custom, all normal DJI start scripts:



And the build.prop file:

```
ro.vendor.product.cpu.abi=arm64-v8a
# begin build properties
# autogenerated by vendor_buildinfo.sh
ro.product.board=evb2
ro.board.platform=eagle2
ro.product.vendor.manufacturer=DJI
ro.product.vendor.model=Android NATIVE on Eagle2 WM265E
ro.product.vendor.brand=eagle2
ro.product.vendor.name=eagle2_wm265e
ro.product.vendor.device=eagle2_wm265e
# end build properties
#
# ADDITIONAL VENDOR BUILD PROPERTIES
#
ro.carrier=unknown
```



As far as could be seen by a quick check, there is nothing custom in this drone firmware. The firmware is a plain stock Mavic 3 Enterprise firmware, we were not able to spot any special Anzu/Aloft software in it. All services that you would expect from an DJI drone were present, none were removed or replaced by own services like you would expect from a custom firmware.

The conclusion on the drone firmware: This is a pretty much default Mavic 3 Enterprise drone firmware. Together with the results of the hardware comparison we can say: On the drone side of things there is no special Anzu/Aloft version to be seen. Maybe we can spot something else in the Controller side of things?



Installed packages on the RC device:

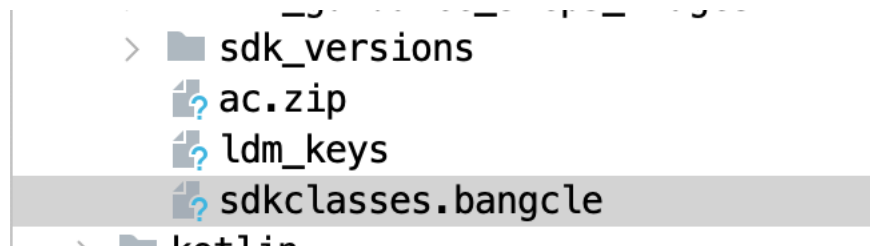
```
package:com.android.internal.display.cutout.emulation.corner
package:com.android.internal.display.cutout.emulation.double
package:com.android.providers.telephony
package:com.android.providers.media
package:com.qti.service.colorservice
package:com.android.internal.systemui.navbar.gestural_wide_b
package:com.android.theme.color.cinnamon
package:com.android.theme.icon_pack.rounded.systemui
package:com.android.documentsui
package:com.android.externalstorage
package:com.android.htmlviewer
package:com.qualcomm.qti.qms.service.connectionsecurity
package:com.android.providers.downloads
package:com.android.networkstack.inprocess
package:com.android.theme.icon_pack.rounded.android
package:vendor.qti.hardware.cacert.server
package:com.android.theme.icon_pack.circular.themepicker
package:com.dpad.update
package:com.android.soundrecorder
package:com.android.providers.downloads.ui
package:ai.aloft.ac_pilot_app
package:android.overlay.common
package:com.android.internal.display.cutout.emulation.tall
package:com.android.modulemetadata
package:com.android.certinstaller
package:com.android.theme.color.black
package:com.android.carrierconfig
package:com.android.theme.color.green
package:com.android.theme.color.ocean
package:com.android.theme.color.space
package:com.android.internal.systemui.navbar.threebutton
package:android
package:com.qualcomm.wfd.service
package:com.android.theme.icon_pack.rounded.launcher
package:com.android.mtp
package:com.android.launcher3
package:com.android.internal.systemui.navbar.twobutton
package:com.android.statementservice
package:com.android.hotspot2
package:com.dpad.devicetest
package:android.overlay.target
package:com.android.internal.systemui.navbar.gestural_extra_wi
package:com.qualcomm.qcrilmsgtunnel
package:com.android.providers.settings
package:com.android.sharedstoragebackup
package:com.android.printspooler
package:com.android.theme.icon_pack.filled.settings
package:com.dpad.full
package:com.android.webview
```

```
package:com.dpad.full
package:com.android.webview
package:com.android.inputdevices
package:com.dpad.service
package:com.android.theme.icon_pack.circular.settings
package:com.android.musicfx
package:com.google.android.webview
package:com.android.theme.icon.teardrop
package:android.ext.shared
package:com.android.keychain
package:com.android.gallery3d
package:com.android.theme.icon_pack.filled.systemui
package:android.ext.services
package:com.android.phone.overlay.common
package:com.qualcomm.qti.qtisystemservice
package:com.android.carrierconfig.overlay.common
package:com.android.systemui.overlay.common
package:com.android.server.telecom.overlay.common
package:com.android.packageinstaller
package:com.android.theme.font.notoserifsource
package:com.android.theme.icon_pack.filled.android
package:com.android.theme.icon_pack.circular.systemui
package:org.mozilla.firefox
package:com.android.theme.icon.squircle
package:com.android.storagemanager
package:com.android.settings
package:com.android.theme.icon_pack.filled.launcher
package:com.android.networkstack.permissionconfig
package:com.android.theme.icon_pack.circular.launcher
package:com.google.android.apps.pdfviewer
package:com.android.vpndialogs
package:com.android.music
package:com.android.phone
package:com.android.shell
package:com.android.theme.icon_pack.filled.themepicker
package:com.android.providers.userdictionary
package:com.qualcomm.qti.qmml
package:com.android.internal.systemui.navbar.gestural
package:com.android.location.fused
package:com.android.theme.color.orchid
package:com.android.systemui
package:com.android.theme.color.purple
package:com.qualcomm.qti.networksetting
package:com.android.permissioncontroller
package:com.qualcomm.qti.qms.service.trustzoneaccess
package:com.dpad.setup
package:com.android.bluetooth
package:com.qualcomm.timeservice
package:com.android.providers.contacts
package:com.android.captiveportallogin
package:com.android.theme.icon.roundedrect
package:com.android.internal.systemui.navbar.gestural_narrow_back
package:com.android.cellbroadcastreceiver.overlay.common
package:com.android.theme.icon_pack.rounded.settings
package:com.google.android.inputmethod.latin
package:com.android.bluetooth.overlay.common
package:com.android.theme.icon_pack.circular.android
```

## The App side of things

Let's take a look at the app, we know by the installed packages list from above that the package name is ai.Aloft.ac\_pilot\_app. Grabbing the APK from the RC allows us to inspect it further, the version we have is 1.12.16.95

First observation is that the app uses the DJI SDK, the SDK itself is protected by secneo/bangle, which is obfuscating and encrypting the original DEX files, so reverse engineering them is hard. Luckily this is not a dealbreaker for us. Due to my good friend Jon the secneo protection is easily removed.



The look into the SDK reveals that this seems to be pretty much a standard SDK from DJI in some older version. We cannot observe any special/custom magic. All functions like DJI cloudcontrol are still in the SDK. If this is an “secure” product, like Anzu claims, it should not use an SDK that has these functions included. Here is the DJI cloudcontrol in the SDK:

```

private String b(b bVar) {
    Set<String> set;
    JSONObject jsonObject = new JSONObject();
    try {
        jsonObject.put("app_name", bVar.a());
        jsonObject.put("env", bVar.d());
        set = this.b;
    } catch (JSONException e) {
        e.printStackTrace();
    }
    if (set != null && set.size() > 0) {
        JSONArray jsonArray = new JSONArray();
        Iterator<String> it = this.b.iterator();
        while (it.hasNext()) {
            jsonArray.put(it.next());
        }
        jsonObject.put("namespaces", jsonArray);
        if (!l.a(this.c)) {
            jsonObject.put("user_id", this.c);
        }
        jsonObject.put("device_uuid", bVar.c());
        jsonObject.put("country", bVar.b());
        jsonObject.put("platform", bVar.e());
        jsonObject.put("platform_version", bVar.f());
        jsonObject.put("app_version", bVar.i());
        if (!l.a(this.a)) {
            jsonObject.put("extra", this.a);
        }
        return jsonObject.toString();
    }
    throw new IllegalArgumentException("namespaces is required");
}

public h a(b bVar) {
    this.d = "https://api.djbservice.org/api/cloudcontrol/config";
    return new h(this.d, c(bVar), b(bVar));
}

```

A little bit of going backwards in the code we find this part of the code to determine if cloudcontrol is used or not:

```

@Override // dji.v5.inner.register.co_a
public boolean co_a() {
    return SDKRelativeJNI.native_isSDKActivated() || LDManager.getInstance().isLDMEabled();
}

```

If LocalDataMode (LDM) is enabled the cloudcontrol service will not run.

Aloft enables LDM with a valid LDMkey:

```

public final String readLicenseContents() {
    try {
        InputStreamReader inputStreamReader = new InputStreamReader(this.context.getAssets().open("ldm_keys/" + this.keyLocation));
        String str = (String) CollectionsKt.firstOrNull((List) TextStreamsKt.readLines(inputStreamReader));
        if (str == null) {
            str = "";
        }
        inputStreamReader.close();
        return str;
    } catch (Exception e) {
        Log.d(TAG, "readLicenseContents: Failed to read license contents: " + e.getLocalizedMessage());
        return "";
    }
}

```

What does that mean? Currently LDM is enabled, but LDM defaults to disabled in the SDK, so if anything unintended happens, LDM will not be enabled and the data will be synced to DJI.

LDM does not disable the internet connection, if any data leaks are stopped by LDM is to be proven separately, in the past LDM still leaked some data depending on the SDK version used. Same goes for the case when DJI decides to void the LDMkey.

It would be much better if Aloft would not rely on the DJI SDK for the app in the first place, as the SDK includes the cloudcontrol function. So instead of disabling it by using LDM it should not be included at all.

Overall, Aloft app is an Pilot app that simply uses the SDK provided by DJI, like any other popular apps for DJI products. It just has some extra functions like warranty activation etc. The Aloft app is not an own rewrite of the DJI fly app, it does not control the hardware on a low level but only speaks to the DJI provided SDK which then sends control commands to the drone via DJI's DUML format.

For claiming security, the app is very sloppy with security itself. By the time looking at the app the staging and production AWS was found in the code.

AWS in the code:

```
public final class FlavorConfig implements IFlavorConfig {
    public static final int $stable = 0;
    public static final FlavorConfig INSTANCE = new FlavorConfig();
    private static final boolean unauthenticatedModeEnabled = true;
    private static final String stagingUpgradeS3Location = "aloft-staging1-pilot-app.s3.us-east-2.amazonaws.com";
    private static final String productionUpgradeS3Location = "aloft-prod1-pilot-app.s3.amazonaws.com";
}
```

Both, staging and prod, AWS S3 can be accessed/downloaded without authentication:

```
$ aws s3 ls aloft-prod1-pilot-app
PRE dji_rc_pro/
PRE raptor_rc_pro/
2024-03-05 10:21:42 168922016 v1.10.3.apk
2024-05-08 15:51:52 140458216 v1.12.16.95.apk
2024-03-29 16:46:05 168442570 v1.12.3.58.apk
2024-04-04 15:20:55 168446609 v1.12.5.65.apk
2024-04-08 09:45:24 168450705 v1.12.6.70.apk
2023-10-31 16:00:11 171976533 v1.7.1.apk
2023-11-09 10:55:21 159603450 v1.7.2.apk
2023-11-09 10:55:21 159623070 v1.7.3.apk
2023-11-09 13:44:33 159468602 v1.7.4.apk
2023-11-13 12:50:44 159515162 v1.7.5.apk
2023-11-09 18:18:21 159517374 v1.7.6.apk
2023-11-13 12:52:40 159525910 v1.7.7.apk
2023-11-13 13:54:54 159526026 v1.7.8.apk
2023-11-30 16:25:10 160788408 v1.7.9.apk
2024-01-16 15:21:07 161360027 v1.8.0.apk
2024-01-16 15:21:07 161522118 v1.8.1.apk
2024-02-02 09:56:53 165520076 v1.8.10.apk
2024-02-06 14:02:17 165541639 v1.8.11.apk
2024-02-12 10:54:34 165526895 v1.8.12.apk
2024-02-14 09:09:06 165533963 v1.8.13.apk
2024-02-15 11:17:15 165537751 v1.8.14.apk
2024-02-21 09:03:52 165565337 v1.8.16.apk
2024-01-16 15:21:07 161598050 v1.8.2.apk
2024-01-16 15:21:07 164008498 v1.8.3.apk
2024-01-19 09:19:36 164063212 v1.8.4.apk
2024-01-19 18:45:07 165501808 v1.8.5.apk
2024-01-22 18:19:14 165501242 v1.8.6.apk
2024-01-24 11:27:47 165473883 v1.8.7.apk
2024-01-26 09:13:27 165470968 v1.8.8.apk
2024-01-31 08:33:32 165520399 v1.8.9.apk
2024-02-26 09:22:40 168955352 v1.9.0.apk
```



- ▼ Aloft\_AWS\_staging\_dump
  - ▼ dji\_rc\_pro
    - 📄 RRC01\_v00.01....0\_20240110.zip
    - 📄 RRC01\_v00.01....11\_20240110.zip
  - ▼ raptor\_rc\_pro
    - 📄 RRC01\_v00.01....9\_20240108.zip
    - 📄 RRC01\_v00.01....0\_20240110.zip
    - 📄 RRC01\_v00.01....1\_20240123.zip
  - 📄 v1.6.0.apk
  - 📄 v1.7.0.apk
  - 📄 v1.7.3-constraints.apk
  - 📄 v1.7.3.apk
  - 📄 v1.7.4.apk
  - 📄 v1.7.5.apk
  - 📄 v1.7.6.apk
  - 📄 v1.7.7.apk
  - 📄 v1.7.8.apk
  - 📄 v1.7.9.apk
  - 📄 v1.7.10.apk
  - 📄 v1.7.11.apk
  - 📄 v1.8.0.apk
  - 📄 v1.8.2.apk
  - 📄 v1.8.3.apk
  - 📄 v1.8.4.apk
  - 📄 v1.8.5.apk
  - 📄 v1.8.6.apk
  - 📄 v1.8.7.apk
  - 📄 v1.8.8.apk
  - 📄 v1.8.9.apk
  - 📄 v1.8.10.apk
  - 📄 v1.8.11.apk
  - 📄 v1.8.12.apk
  - 📄 v1.8.13.apk
  - 📄 v1.8.14.apk
  - 📄 v1.8.16.apk
  - 📄 v1.9.0.apk
  - 📄 v1.9.1.apk
  - 📄 v1.9.3.apk
  - 📄 v1.10.3.21.apk
  - 📄 v1.11.1.25.apk
  - 📄 v1.12.0.48.apk
  - 📄 v1.12.1.55.apk
  - 📄 v1.12.3.57.apk
  - 📄 v1.12.4.61.apk
  - 📄 v1.12.5.64.apk
  - 📄 v1.12.6.69.apk
  - 📄 v1.12.7.71.apk
  - 📄 v1.12.8.72.apk

When claiming to be certified:

#### ^ HOW IS MY DATA STORED WHEN CAPTURED WITH THE RAPTOR-SERIES DRONES?

During image or video capture, files are stored on the drone's onboard SD card. Upon landing, you can easily remove the SD card and transfer content to other devices for viewing and editing. Flight logs, accessible via drone connectivity, serve for warranty claims and other analyses initiated through standard service processes, such as an RMA or service request.

All servers used by Anzu Robotics are through our strategic software partner Aloft and are US-based. Your data remains secure, as we do not transmit any information to third parties unless explicitly transferred by the drone owner.

Aloft is ISO 27001 and SOC 2, Type II certified.

This should not happen...

Please look at:

<https://www.anzurobotics.com/data-security/>

More precise:

<https://www.anzurobotics.com/wp-content/uploads/2024/04/Anzu-Robotics-Cyber-Disclosure-2024.pdf>

This are very strong claims, especially "Aloft holds ISO 27001 and SOC 2 TYPE II certifications, demonstrating their commitment to information security and adherence to rigorous security practices."

Leaving an AWS open without noticing that several researchers already dumped the contents of both staging and production does not build trust.

(Sidenote, they have an security paper, go for it for a fun read:

<https://www.aloft.ai/wp-content/uploads/2021/05/Aloft-Netsec-White-Paper-Final-05.17.21.pdf>)

Interesting fact: The AWS share included an DJI RC Pro folder, with the same firmware version as the RRC01 (Raptor Remote Controller 01) firmware, the RRC01 firmware is just copy & paste of the rm510 (DJI RC Pro) firmware with the swapped app ...

Let's wrap up ...

Anyway, Anzu tried to position itself as a drone company using licensed technology by DJI, in an "secured" way.

The analysis of the drone which we were able to access shows that these claims are far exaggerated, the drone is just a green painted Mavic 3 Enterprise, and the RC comes with another app bundled that is nothing but a flying app using the DJI SDK with some custom configuration. We were not able to find any proof of "custom" drone firmware, the only custom part was the app, but the app still heavily relies on DJI technology, most likely any other DJI flying app can be used if run on the controller.

Conclusion: The Anzu Raptor is, just a green DJI Mavic 3 Enterprise, without any substantial own development. The RRC01 is just an RC Pro painted green with a swapped app, which is just another UI for the DJI SDK. Overall, we feel the meme on the cover page describes the situation best.