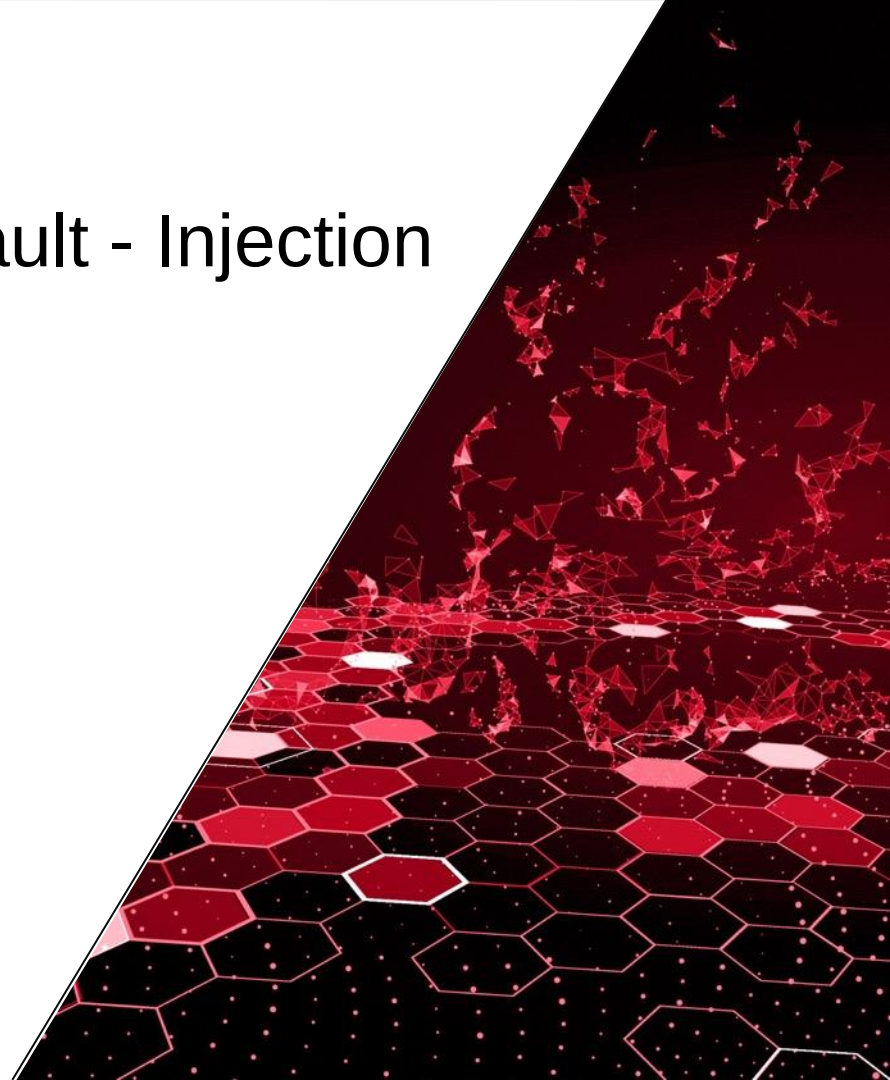
Setting up Side-channel / Fault - Injection Attacks on UAVs

John Sheehy
SVP, Research and Strategy

IOActive®



IOActive Presentation Content

Legal Notices

- **Disclaimer Notification**
  The views, opinions, findings, conclusions, positions, and/or recommendations expressed herein are those of the authors individually and do not necessarily reflect the views, opinions, or positions of IOActive, Inc.
- **No Warranties or Representations**
  The information presented herein is provided “AS IS” and IOActive disclaims all warranties whatsoever, whether express or implied. Further, IOActive does not endorse, guarantee, or approve, and assumes no responsibility for nor makes any representations regarding the content, accuracy, reliability, timeliness, or completeness of the information presented. Users of the information contained herein assume all liability from such use.
- **Publicly Available Material**
  All source material referenced in this presentation was obtained from the Internet without restriction on use.
- **Fair Use**
  This primary purpose of this presentation is to educate and inform. It may contain copyrighted material, the use of which has not always been specifically authorized by the copyright owner. We are making such material available in our efforts to advance understanding of cyber safety and security. This material is distributed without profit for the purposes of criticism, comment, news reporting, teaching, scholarship, education, and research, and constitutes fair use as provided for in section 107 of the Copyright Act of 1976.
- **Trademarks**
  IOActive, the IOActive logo and the hackBOT logo are trademarks and/or registered trademarks of IOActive, Inc. in the United States and other countries. All other trademarks, product names, logos, and brands are the property of their respective owners and are used for identification purposes only.
- **No Endorsement or Commercial Relationship**
  The use or mention of a company, product or brand herein does not imply any endorsement by IOActive of that company, product, or brand, nor does it imply any endorsement by such company, product manufacturer, or brand owner of IOActive. Further, the use or mention of a company, product, or brand herein does not imply that any commercial relationship has existed, currently exists, or will exist between IOActive and such company, product manufacturer, or brand owner.
- **Copyright**

Briefer

John Sheehy

SVP, Research and Strategy

Bio: John has overseen multiple projects delivering identity management, threat modeling, industrial control systems security, risk assessment, security policy, secure device design, and incident & breach simulation and response services. His experience includes over 20 years of system architecture, systems integration, and information security experience working in Enterprise Architecture, Identity & Access Management, Vulnerability & Threat Management, Operations Technology, Security Strategy, Systems Architecture, and Hardware/Application Security domains.

He currently leads IOActive's research program, corporate strategy, and service offering development. He spent two years as Acting CISO of IOActive.

john.sheehy@ioactive.com

IOActive.

Lead Researcher

Gabriel Gonzalez Garcia

Director of Hardware Security

Bio: Gabriel has completed hundreds of projects involving reverse engineering, code review, integrated hardware and software penetration testing. His areas of focus in technologies are hardware and embedded systems. However, he has special interests in low-level attack vectors including fault-injection and side-channel analysis and also exploring novel attack pathways. He applies these skills to work in the automotive, aircraft, smart grid, SATCOM, ATM, IOT, and utility industries in addition to others.

Gabriel also has presented a considerable amount of original cybersecurity research at major conferences including Black Hat Europe. He's passionate about making our cyber world safer.

gabriel.gonzalez@ioactive.com

IOActive.

Bottomline Up Front

Gabriel was able to get code execution on target drone using fault injection attack vector

Target Drone had a strong cybersecurity posture

Code execution allows for modification of firmware image

Attack demonstrates potential consequences for embedded systems

IOActive.

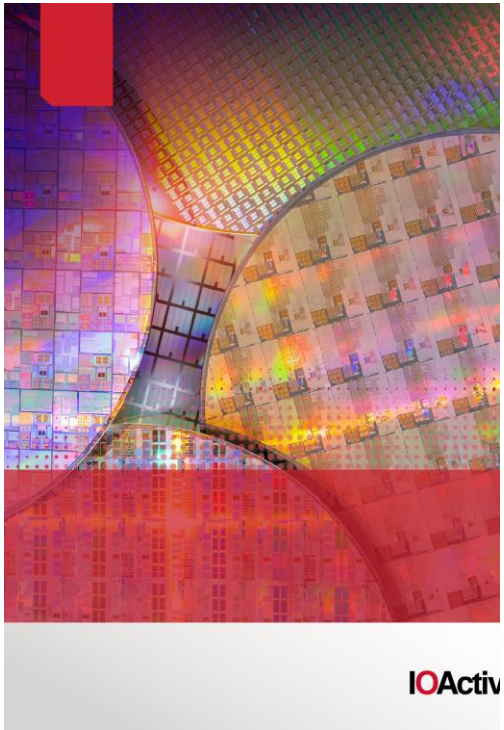
Background: Project

- Project Timeline
  - Started in 2022
  - Completed Responsible Disclosure in May 2023
  - Results published in June 2023

- Blogs
  - https://ioactive.com/drone-security-fault-injection-attacks-gabriel-gonzalez/
  - https://ioactive.com/applying-fault-injection-to-the-firmware-update-process-of-a-drone/
  - https://ioactive.com/hed-hack-the-sky-adventures-in-drone-security-gabriel-gonzalez/

- Whitepaper
  - https://ioactive.com/drone-security-fault-injection-attacks-gabriel-gonzalez/

Background: Low-level Hardware Attacks



- Blog
  - https://ioactive.com/threat-brief-low-level-hardware-attacks/
- eGuide
  - https://info.ioactive.com/acton/fs/blocks/showLandingPage/a/34793/p/p-009c/t/page/fm/0

IOActive.

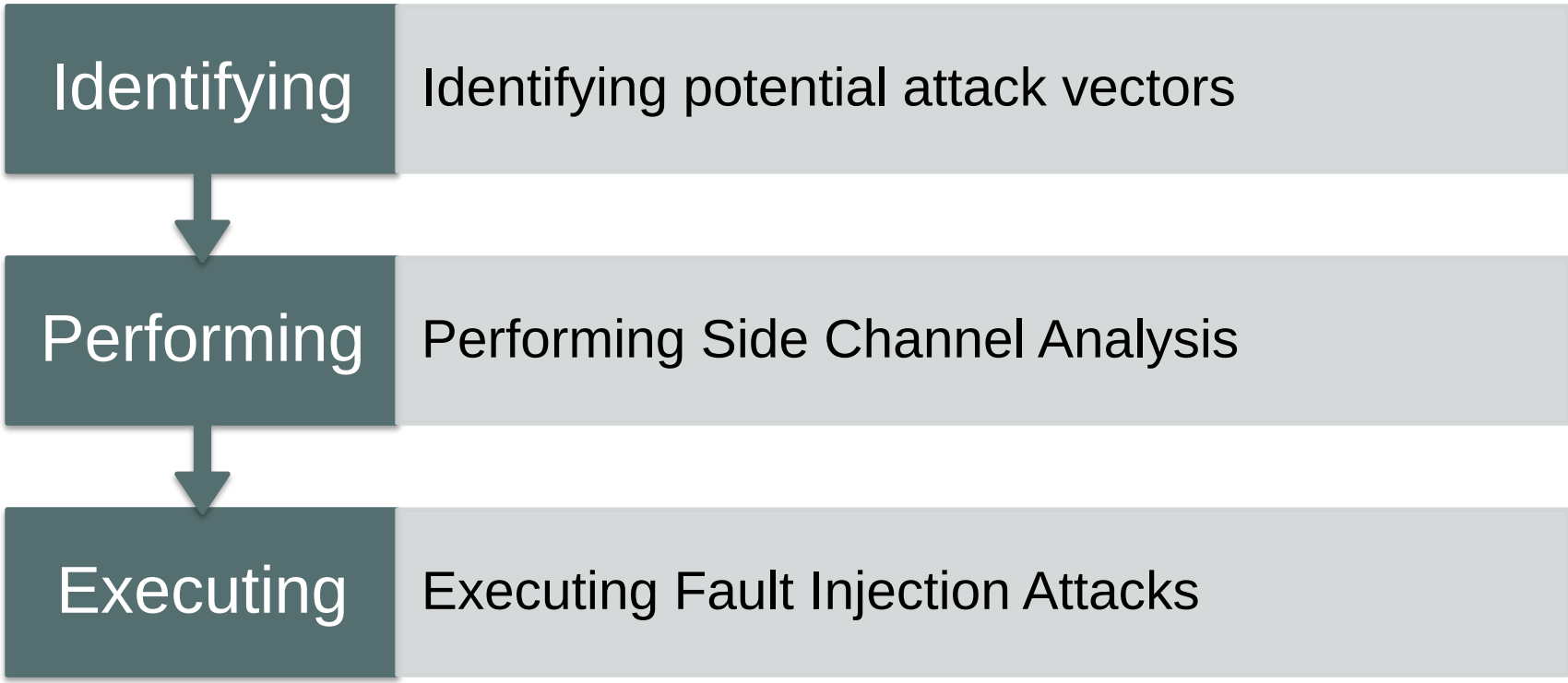
Goals of the Project

- Break security mechanisms of commercially available Drones
- Use a target device with good security posture
- Demonstrate code execution and/or leakage of cryptographic secrets

What was achieved?

- Side-Channel Attacks against Firmware Upgrade is likely unfeasible
- Code Execution demonstrated using Fault Injection against Firmware Upgrade Process
- No Drones were hurt during the process

Project stages

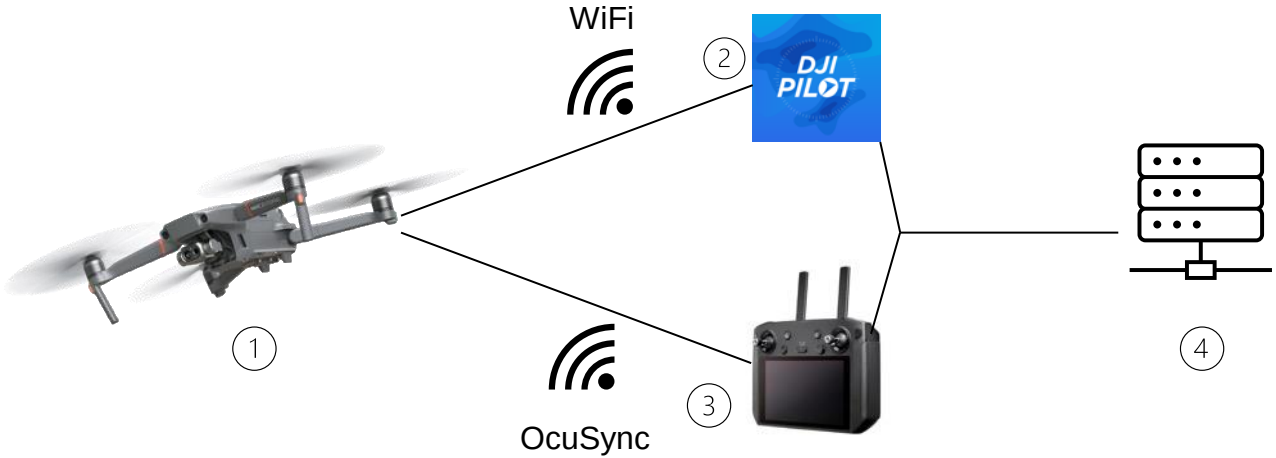


IOActive.

Security of a UAV

- Drones/UAV are becoming increasingly more important
- Varying levels of security
- Latest versions of top brands have improved security
- Target for this project: DJI Drones

Attack Vectors



IOActive.

OSINT

Online community around DJI Drones	https://github.com/o-gs/dji-firmware-tools
Mattermost channel	https://drone-hacks.com/birdmap

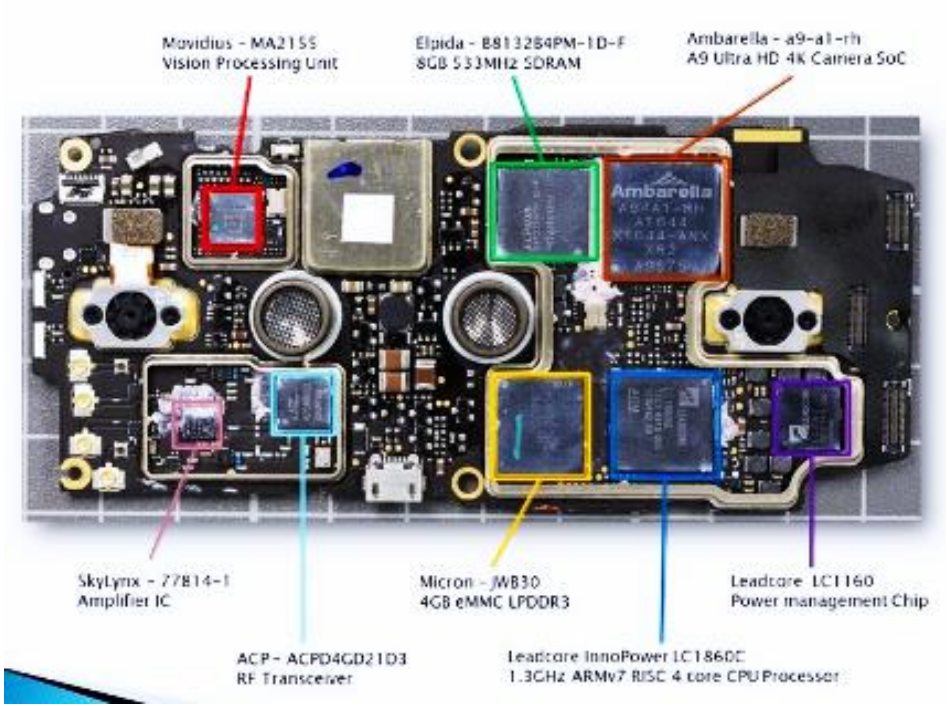
Setting up Testing environment

Identify the right model

Debugging access is key in this stage

It will help moving faster: BlackBox vs WhiteBox

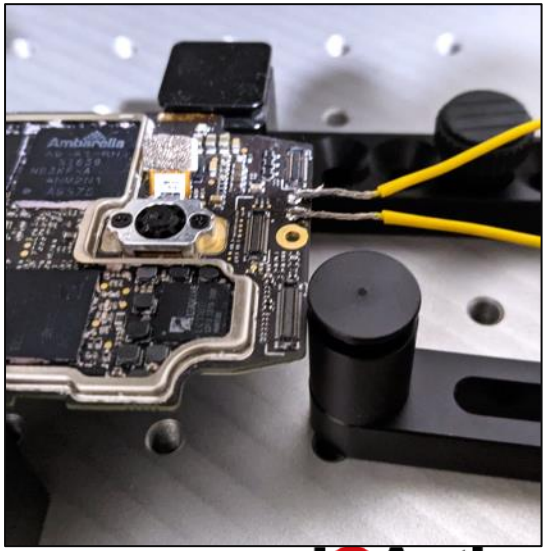
Setting up the Testing Environment



IOActive.

Setting up Testing environment

- Make the PCB run as stand-alone device
- Problem: The unit reboots every few seconds
- Solution: Big fan



IOActive.

Identifying Assets

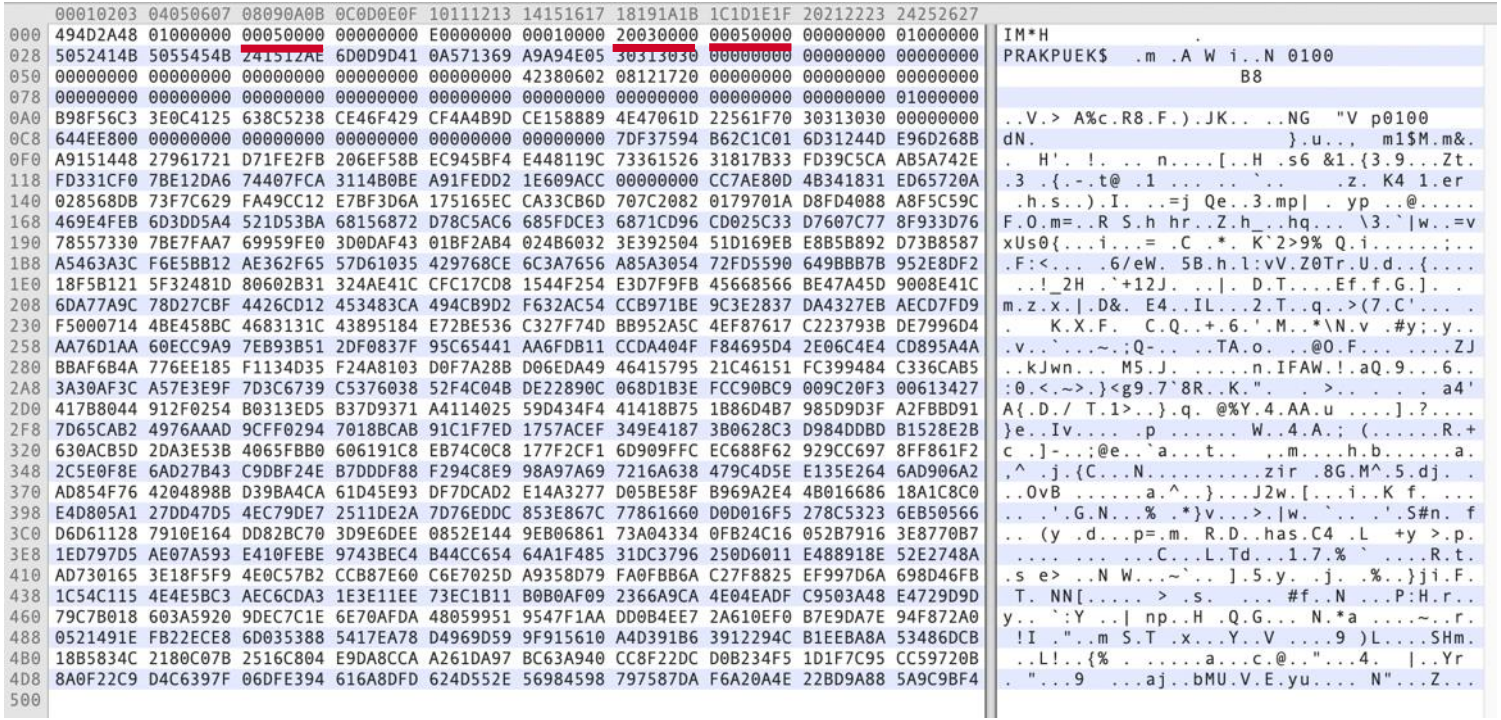
The goal is to get access to the latest firmware

Distributed firmware files are signed and encrypted

Study feasibility of applying side-channel attacks

IOActive.

Setting up Testing environment



```
    00010203 04050607 08090A0B 0C0D0E0F 10111213 14151617 18191A1B 1C1D1E1F 20212223 24252627
000 494D2A48 01000000 00050000 00000000 E0000000 00010000 20030000 00050000 00000000 01000000
028 5052414B 5055454B 24151ZAE 6D0D9D41 0A571369 A9A94E05 30313030 00000000 00000000 00000000
050 00000000 00000000 00000000 00000000 00000000 42380602 08121720 00000000 00000000 00000000
078 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 01000000
0A0 B98F56C3 3E0C4125 638C5238 CE46F429 CF4A4B9D CE158889 4E47061D 22561F70 30313030 00000000
0C8 644EE800 00000000 00000000 00000000 00000000 00000000 7DF37594 B62C1C01 6D31244D E96D268B
0F0 A9151448 27961721 D71FE2FB 206EF58B EC945BF4 E448119C 73361526 31817B33 FD39C5CA AB5A742E
118 FD331CF0 7BE12DA6 74407FCA 3114B0BE A91FEDD2 1E609ACC 00000000 CC7AE80D 4B341831 ED65720A
140 028568DB 73F7C629 FA49CC12 E7BF3D6A 175165EC CA33CB6D 707C2082 0179701A D8FD4088 A8F5C59C
168 469E4FEB 6D3DD5A4 521D53BA 68156872 D78C5AC6 685FDCE3 6871CD96 CD025C33 D7607C77 8F933D76
190 78557330 7BE7FAA7 69959FE0 3D0DAF43 01BF2AB4 024B6032 3E392504 51D169EB E8B5B892 D73B8587
1B8 A5463A3C F6E5BB12 AE362F65 57D61035 429768CE 6C3A7656 A85A3054 72FD5590 649BBB7B 952E8DF2
1E0 18F5B121 5F32481D 80602B31 324AE41C CFC17CD8 1544F254 E3D7F9FB 45668566 BE47A45D 9008E41C
208 6DA77A9C 78D27CBF 4426CD12 453483CA 494CB9D2 F632AC54 CCB971BE 9C3E2837 DA4327EB AECD7FD9
230 F5000714 4BE458BC 4683131C 43895184 E72BE536 C327F74D BB952A5C 4EF87617 C223793B DE7996D4
258 AA76D1AA 60ECC9A9 7EB93B51 2DF0837F 95C65441 AA6FDB11 CCDA404F F84695D4 2E06C4E4 CD895A4A
280 BBAF6B4A 776EE185 F1134D35 F24A8103 D0F7A28B D06EDA49 46415795 21C46151 FC399484 C336CAB5
2A8 3A30AF3C A57E3E9F 7D3C6739 C5376038 52F4C04B DE22890C 068D1B3E FCC90BC9 009C20F3 00613427
2D0 417B8044 912F0254 B0313ED5 B37D9371 A4114025 59D434F4 41418B75 1B86D4B7 985D9D3F A2FBBD91
2F8 7D65CAB2 4976AAAD 9CFF0294 7018BCAB 91C1F7ED 1757ACEF 349E4187 3B0628C3 D984DDBD B1528E2B
320 630ACB5D 2DA3E53B 4065FBB0 606191C8 EB74C0C8 177F2CF1 6D909FFC EC688F62 929CC697 8FF861F2
348 2C5E0F8E 6AD27B43 C9DBF24E B7DDDF88 F294C8E9 98A97A69 7216A638 479C4D5E E135E264 6AD906A2
370 AD854F76 4204898B D39BA4CA 61D45E93 DF7DCAD2 E14A3277 D05BE58F B969A2E4 4B016686 18A1C8C0
398 E4D805A1 27DD47D5 4EC79DE7 2511DE2A 7D76EDDC 853E867C 77861660 D0D016F5 278C5323 6EB50566
3C0 D6D61128 7910E164 DD82BC70 3D9E6DEE 0852E144 9EB06861 73A04334 0FB24C16 052B7916 3E8770B7
3E8 1ED797D5 AE07A593 E410FEBE 9743BEC4 B44CC654 64A1F485 31DC3796 250D6011 E488918E 52E2748A
410 AD730165 3E18F5F9 4E0C57B2 CCB87E60 C6E7025D A9358D79 FA0FBB6A C27F8825 EF997D6A 698D46FB
438 1C54C115 4E4E5BC3 AEC6CDA3 1E3E111E 73EC1B11 B0B0AF09 2366A9CA 4E04EADF C9503A48 E4729D9D
460 79C7B018 603A5920 9DEC7C1E 6E70AFDA 48059951 9547F1AA DD0B4EE7 2A610EF0 B7E9DA7E 94F872A0
488 0521491E FB22ECE8 6D035388 5417EA78 D4969D59 9F915610 A4D391B6 3912294C B1EEBA8A 53486DCB
4B0 18B5834C 2180C07B 2516C804 E9DA8CCA A261DA97 BC63A940 CC8F22DC D0B234F5 1D1F7C95 CC59720B
4D8 8A0F22C9 D4C6397F 06DFE394 616A8DFD 624D552E 56984598 797587DA F6A20A4E 22BD9A88 5A9C9BF4
500

IM*H              .
PRAKPUEK$  .m .A W i..N 0100
                    B8
                                                    
..V.> A%c.R8.F.).JK.. ..NG  "V p0100
dN.                          }.u..,  m1$M.m&.
.  H'. !. .. n....[..H .s6 &1.{3.9...Zt.
.3 .{.-.t@ .1 ... .. `..     .z. K4 1.er
 .h.s..).I. ..=j Qe..3.mp| . yp ..@.....
F.O.m=..R S.h hr..Z.h_..hq... \3.`|w..=v
xUs0{...i...= .C .*. K`2>9% Q.i......;..
.F:<...  .6/eW. 5B.h.l:vV.Z0Tr.U.d..{....
 ..!_2H .`+12J. ..|. D.T....Ef.f.G.]. .
m.z.x.|.D&. E4..IL...2.T..q..>(7.C'... .
.   K.X.F.  C.Q..+.6.'.M..*\N.v .#y;.y..
.v..`...~.;Q-.. ..TA.o. ..@O.F... ....ZJ
..kJwn... M5.J. .....n.IFAW.!.aQ.9...6..
:0.<.~>.}<g9.7`8R..K.".  . >.. . . . a4'
A{.D./ T.1>..}.q. @%Y.4.AA.u ....].?....
}e..Iv.... .p ..... W..4.A.; (......R.+
c .]-..;@e..`a...t..  ,.m....h.b......a.
.^ .j.{C...N..........zir .8G.M^.5.dj. .
..OvB ......a.^..}...J2w.[...i..K f. ...
.. .'.G.N...% .*}v...>.|w. `.. .'.S#n. f
.. (y .d...p=.m. R.D..has.C4 .L  +y >.p.
 .... ... ...C...L.Td...1.7.% ` ....R.t.
.s e> ..N W...~`.. ].5.y. .j. .%..}ji.F.
 T. NN[..... > .s.  ... #f..N ...P:H.r..
y.. `:Y ..| np..H .Q.G... N.*a ....~..r.
 !I ."..m S.T .x...Y..V ....9 )L....SHm.
 ..L!..{% . .....a...c.@.."...4.  |..Yr
. "...9  ...aj..bMU.V.E.yu.... N"...Z...
```


IOActive.

Setting up Testing environment

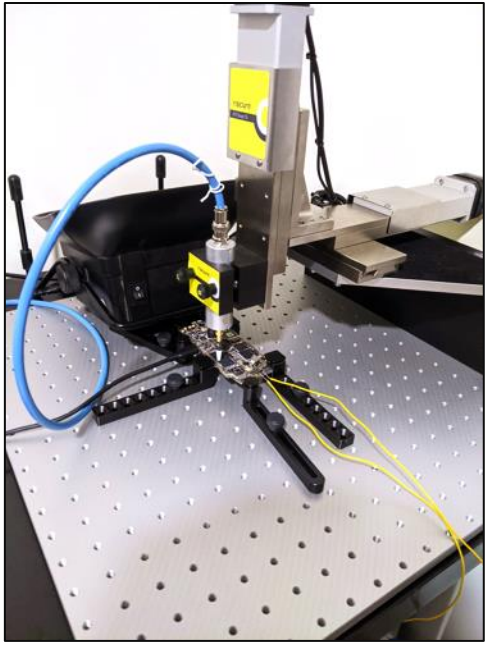
- Identify the right binary
- Signature verification
- Create custom upgrade packages

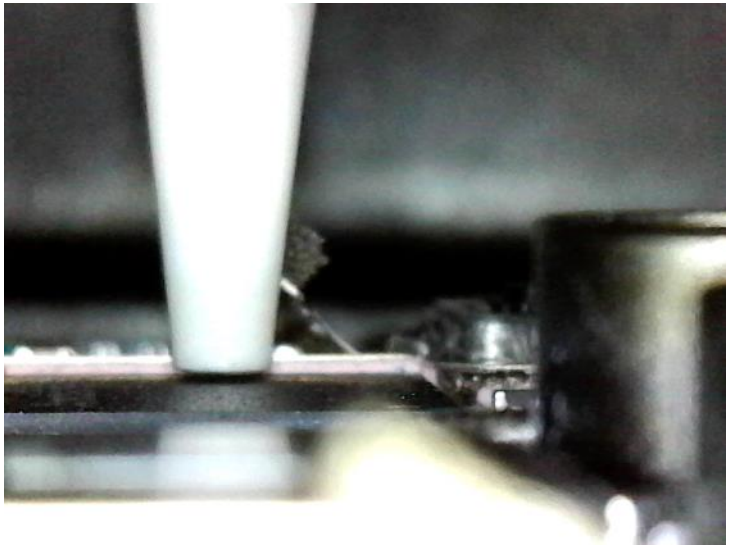
Setting up Testing environment

- Side Channel Analysis
  - Power Consumption Leakage
  - EM Radiation Leakage

Side Channel with Inspector

- Setting XYZ station





IOActive.

Side Channel with Inspector

- Trace generation setup on controlled environment

① Generate Random Package

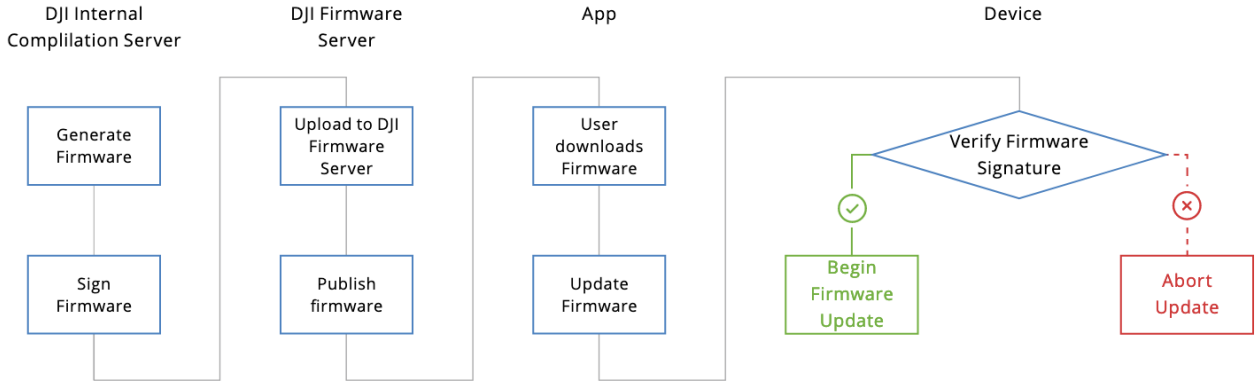
② Initiate Firmware Update

③ Record Data

IOActive.

Side Channel: Conclusion

- What else is needed to success in real life?



IOActive.

Side Channel with Inspector

- Trace generation setup on a real environment

1 Generate Random Package

2 Initiate Firmware Update

3 Bypass Signature

4 Record Data

IOActive.

Second Approach

- Focus on the software update process
- Study Riscure’s trick to alter instructions
- Execute Attack on our target Drone

Analyze Approach

- Targeting Software update mechanism
- Copies hundreds of MegaBytes over USB
- Large number of Load and Store Operation

All systems copy data from A to B! riscure

Single word copy using LDR / STR

```
1        WordCopy:
2           LDR r3, [r1], #4
3           STR r3, [r0], #4
4           SUBS r2, r2, #4
5           BGE WordCopy
```

Multi-word copy using LDMIA / STMIA

```
1        MultiWorldCopy:
2           LDMIA r1!, {r3 - r10}
3           STMIA r0!, {r3 - r10}
4           SUBS r2, r2, #32
5           BGE MultiWorldCopy
```

Public 6

Corrupting load instructions to control PC riscure

Controlling PC using LDR

```
LDR r3, [r1], #4        11100100100100010011000000000100

LDR PC, [r1], #4        11100100100100011111000000000100
```

Controlling PC using LDMIA

```
LDMIA r1!,{r3-r10}      11101000101100010000011111111000

LDMIA r1!,{r3-r10,PC}   11101000101100011000011111111000
```


IOActive.

Setting up Testing environment

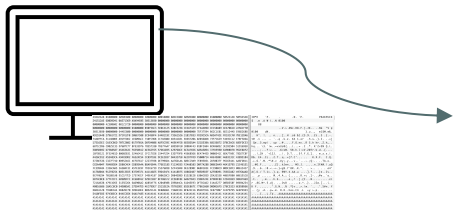
IOActive.

Fault Injection Attack

IOActive.

Fault Injection Attack

- Fault Injection Setup



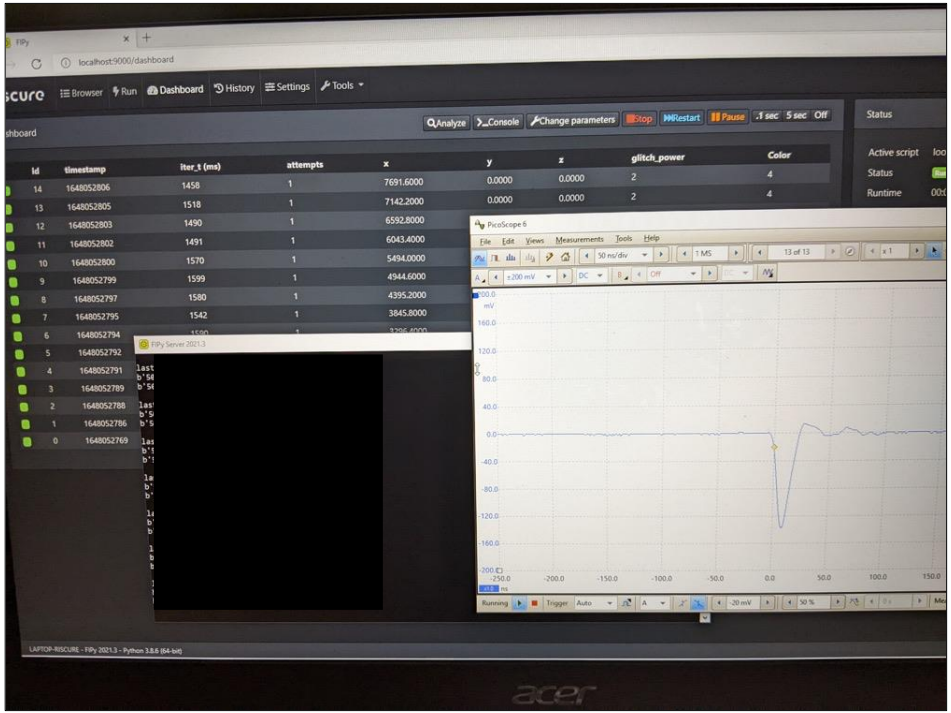
① 0x41414141 Package

② Initiate Firmware Update

③ Continuos Glitching

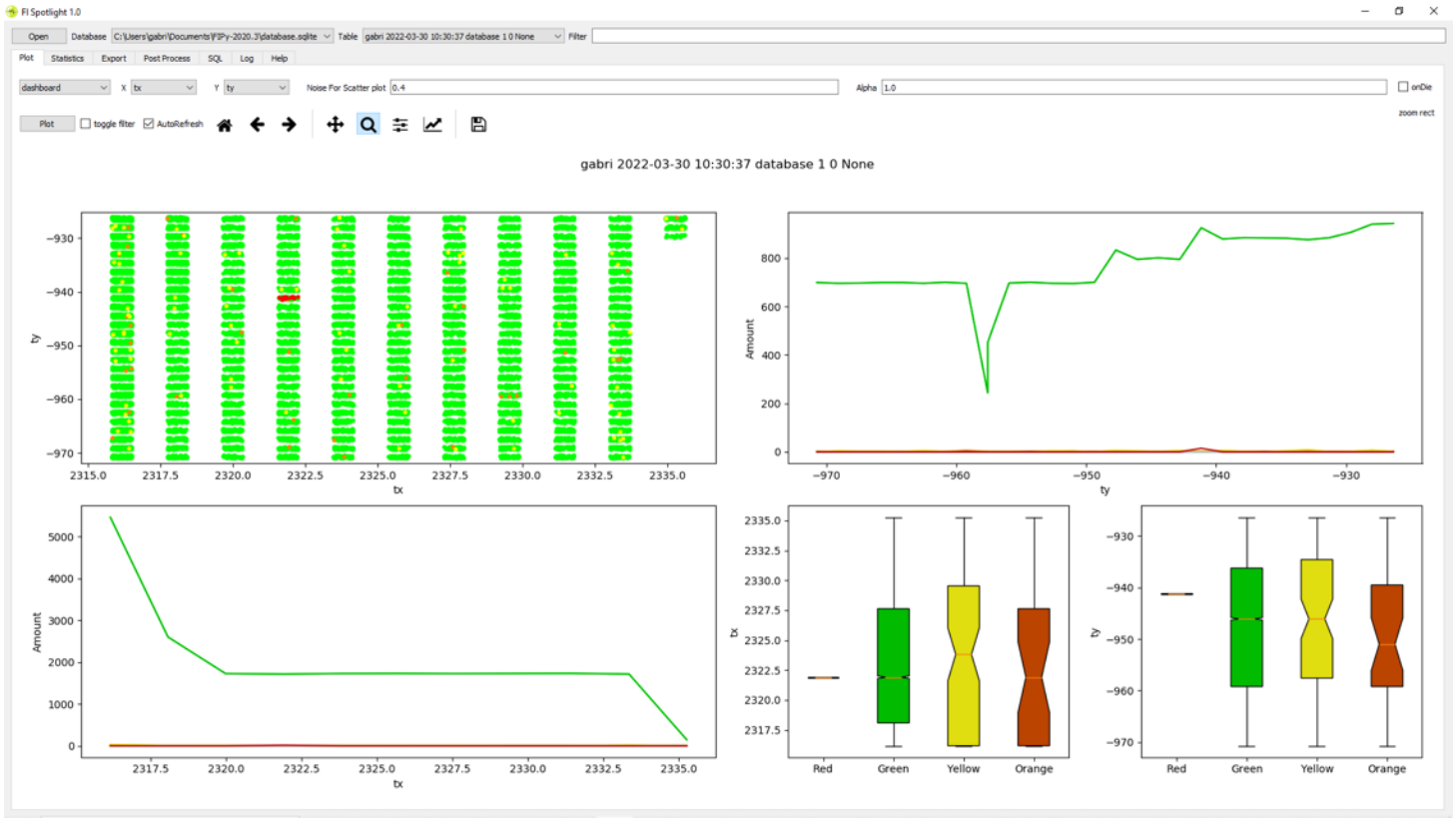
IOActive.

Fault Injection with FiPy



IOActive.

Fault Injection with FiPy



IOActive.

Fault Injection Attack

- After a couple of weeks of trials an exploitable glitch was recorded
- Using GDB it was easier to debug the corrupted code

```
Program received signal SIGSEGV, Segmentation fault.
0x2a002f44 in ?? ()
#0  0x2a002f44 in ?? ()
#1  0x2a000f5a in ?? ()
#2  0x2a0011de in ?? ()
#3  0x2a000b04 in ?? ()
#4  0xb6f9b42c in __libc_init () from /system/lib/libc.so
#5  0x2a00099c in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)
r0             0x41414141  1094795585
r1             0x41414141  1094795585
r2             0xbefff6dc  3204445916
r3             0x41414141  1094795585
r4             0x41414141  1094795585
r5             0xbefffbc4  3204447172
r6             0x85242d9a  2233740698
r7             0xbefff6f0  3204445936
r8             0xb6b4e008  3065307144
r9             0xb6b4e1e8  3065307624
r10            0xbefffad0  3204446928
r11            0x2d7160  2978144
r12            0xbefff6ec  3204445932
sp             0xbefff6c8  0xbefff6c8
lr             0xde  222
pc             0x2a002f44  0x2a002f44
cpsr           0x60000030  1610612784
```


IOActive.

Fault Injection Attack

- Original Code

```
loc_2F40                                ; CODE
                MOV             R7, R2
                LDM             R7!, {R0,R1}
                STR             R0, [R3]
                STR             R1, [R3,#4]
                ADDS            R3, #8
                CMP             R7, R12
                MOV             R2, R7
                BNE             loc_2F40
```

- State of the registers after successful glitch

r0	0x41414141	1094795585
r1	0x41414141	1094795585
r2	0xbefff6dc	3204445916
r3	0x41414141	1094795585
r4	0x41414141	1094795585

Fault Injection Attack

- The original instruction encoding is:

```
LDM R7!, {R0, R1}                    0300B7E8
```

- Potential modified instruction:

```
LDM R7!, {R0, R1, R3, R4}            1B00B7E8
```

- The glitch was able to modify one byte by flipping two consecutive bits

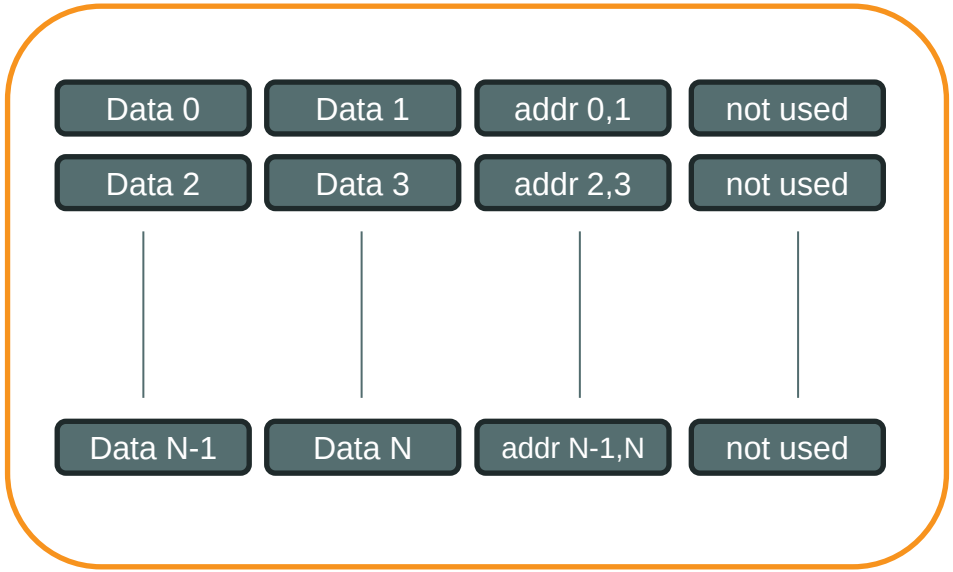
Fault Injection Attack

- Modified code would look like this

```
Loc_2F40
        MOV             R7, R2
        LDM             R7!, {R0, R1, R3, R4}
        STR             R0, [R3]
        STR             R1, [R3, 4]
        ADDS            R3, #8
        CMP             R7, R12
        MOV             R2, R7
        BNE             loc_2F40
```

Fault Injection Attack

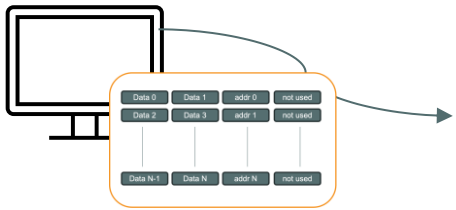
- Payload would look like this



IOActive.

Fault Injection Attack

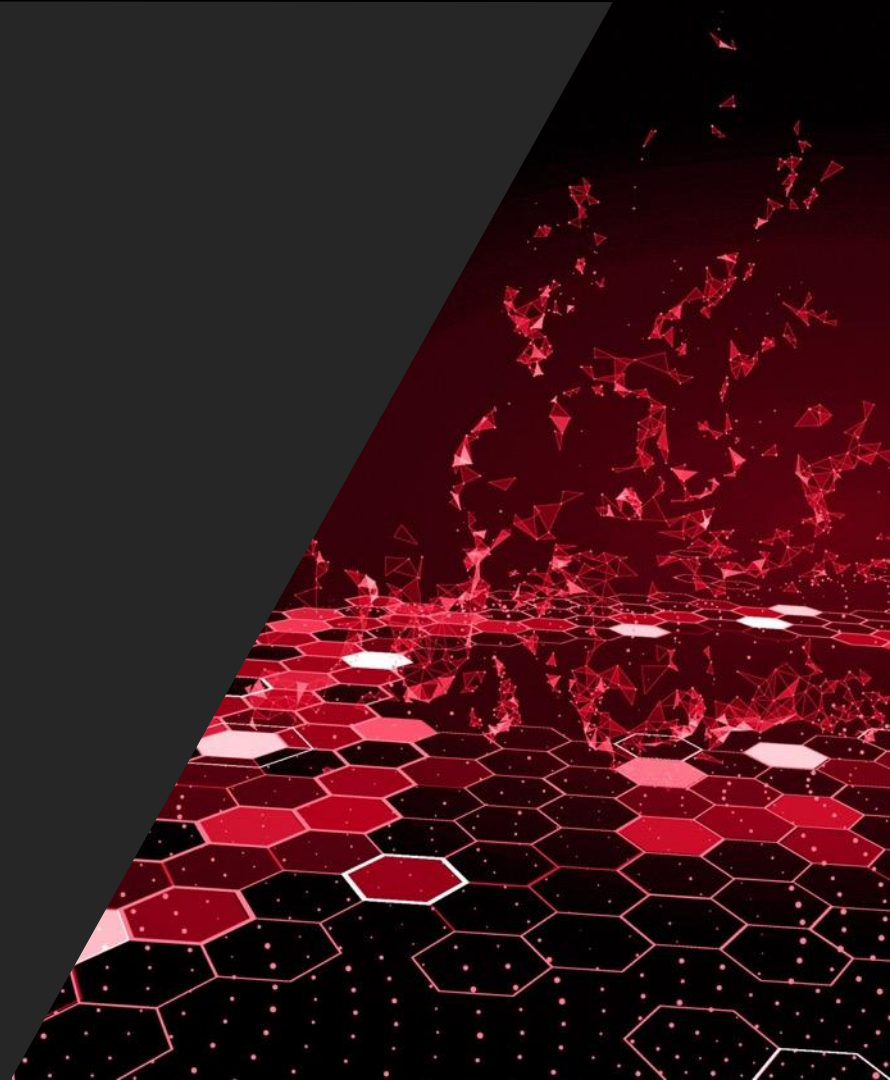
- Attack Flow



1. Custom Payload Package
2. Initiate Firmware Update
3. Continuos Glitching

IOActive.

Questions



IOActive®