DROP (DRone Open source Parser) Forensic analysis of the DJI Phantom III

Devon Clark, Christopher Meffert, Ibrahim Baggili, Frank Breitinger
University *of* New Haven
Cyber Forensics Research and Education Group

University *of* New Haven
Cyber Forensics Research & Education Group

Agenda

UNHcFREG

- Introduction
- Contribution
- Related work
- Methodology
- Analysis
- Results
- Future work
- Final Words
- Questions

Intro: Motivation

- Drones have become more popular in recent years
- Several crimes have been committed using drones
- 2.4 million hobbyist drones purchased in the U.S. in 2016

Source: https://ifunny.co/fun/O2bAr6S63?gallery=tag&query=drone Source: http://fulldrone.com/drone-guns-legal/

Intro: Relevance - DJI

UNHcFREG

- One of the most prominent consumer drone manufacturers today
- $1.5 Billion valuation (USD, 2016)
- DJI held 50% of the North American consumer drone market share in 2016
- Already used by ISIS

Source: https://www.recode.net/2017/4/14/14690576/drone-market-share-growth-charts-dji-forecast

Intro: DJI Phantom III Standard

UNHcFREG

- Four main drone models – Inspire, **Phantom**, Mavic, Spark
- The drone communicates to the remote via proprietary wireless protocol
- The remote acts as a Wi-Fi access point
- A phone can be connected to the remote's Wi-Fi
  - DJI GO application is used for takeoff and landing, taking pictures and video, and serves as a dashboard

Contribution

- Set of procedures that examiners can follow during the investigation of a case that involves a DJI Phantom III Standard drone
- First open account for the binary DAT file structure
- Created first open source Python tool called DRone Open source Parser (DROP) capable of parsing the proprietary DAT file format in a forensically sound manner
- First method for correlating data extracted from the drone's internal storage and the mobile device used to control it

Related Work

- Major points from Unmanned aerial vehicles: A preliminary analysis of forensic challenges by Graeme Horsman
  - Parrot Bebop UAV with iPhone 6 and Galaxy S3, both using the Parrot's application ‘FreeFlight3’.
  - Acquired dd image of internal storage via telnet and access other system files through ftp
  - The drone has no security for its wifi: no password. Also, flight data is not encrypted
  - Do not use no fly zone geo-fencing
- UAV Forensics – David Kovar – March 2015
  - Forensic analysis of DJI Phantom 2
- DEFCON 2016 – Aaron Luo
  - Detailed security analysis of the DJI Phantom III

Methodology

UNHcFREG

Factory reset
- Ensure no external variables affected our results.

Scenario creation
- Powered on, and flown at different locations.

Data acquisition
- Drone
- Controller
- Tablet

Data analysis
- Txt file
- DAT files

Tool creation
- DROP python tool was constructed

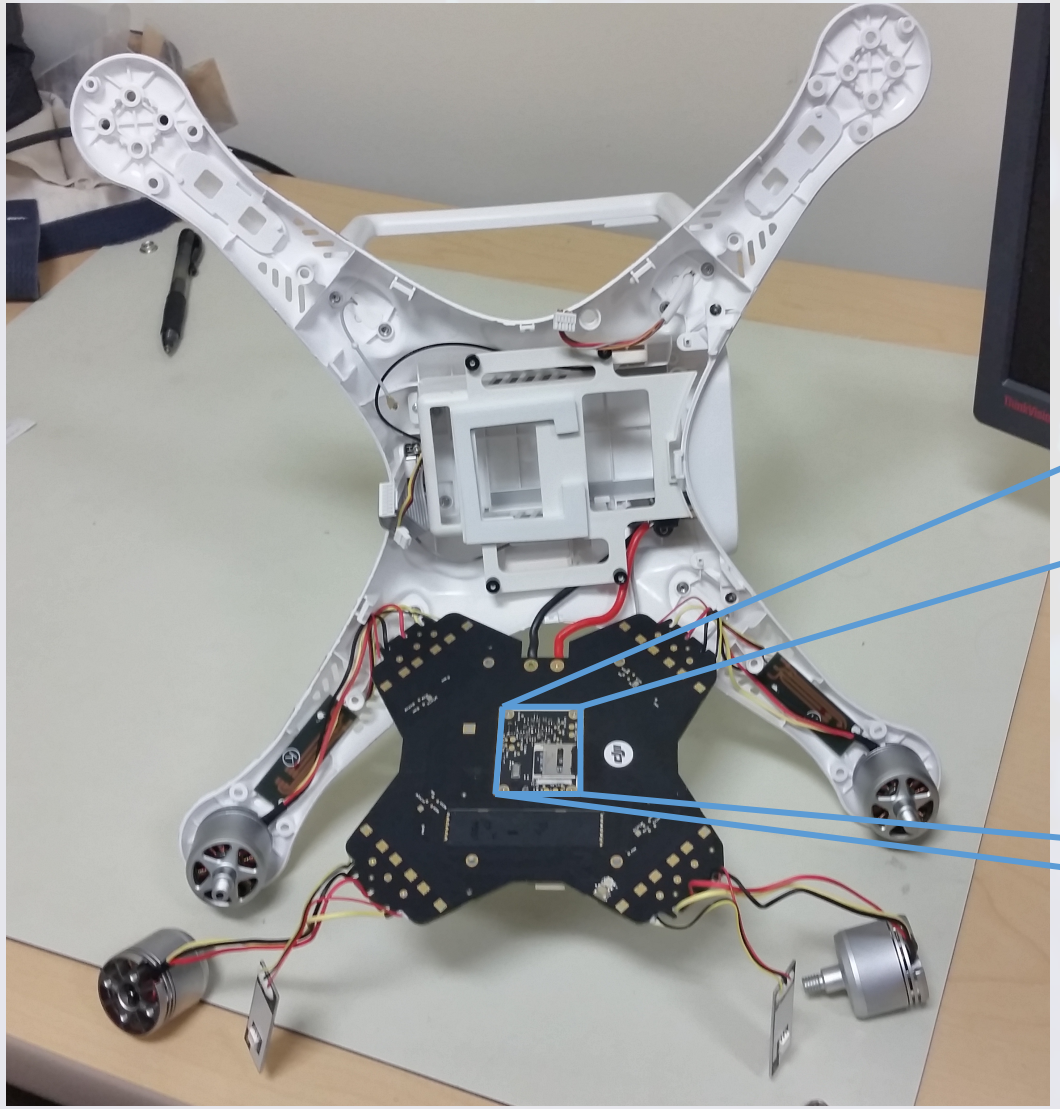
Testing
- Carried out to validate findings

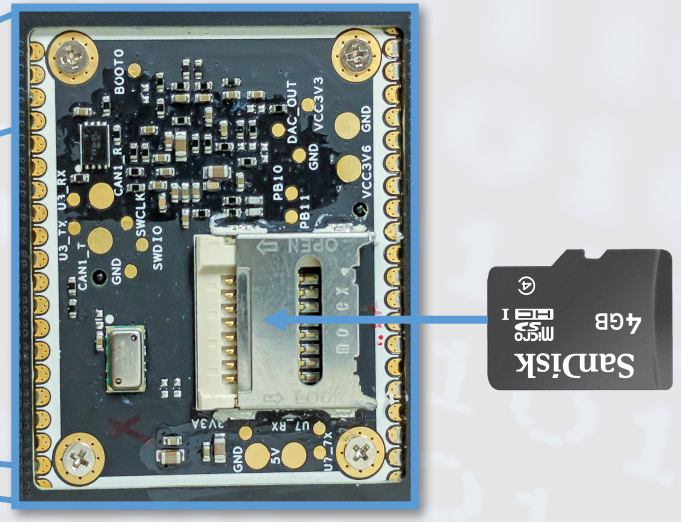
Methodology: Data Acquisition

- Android backup
- Manual acquisition of Android internal storage
- Imaging of the drone’s internal SD card
- Imaging of Gimbal SD card

Methodology: Disassembly

UNHcFREG



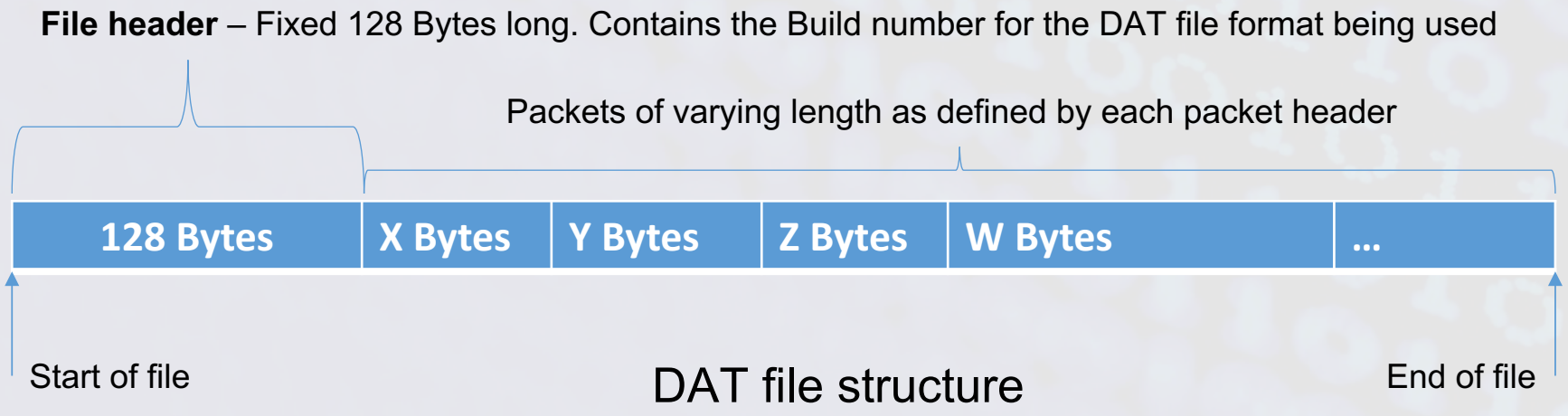


Analysis: Where is data stored?

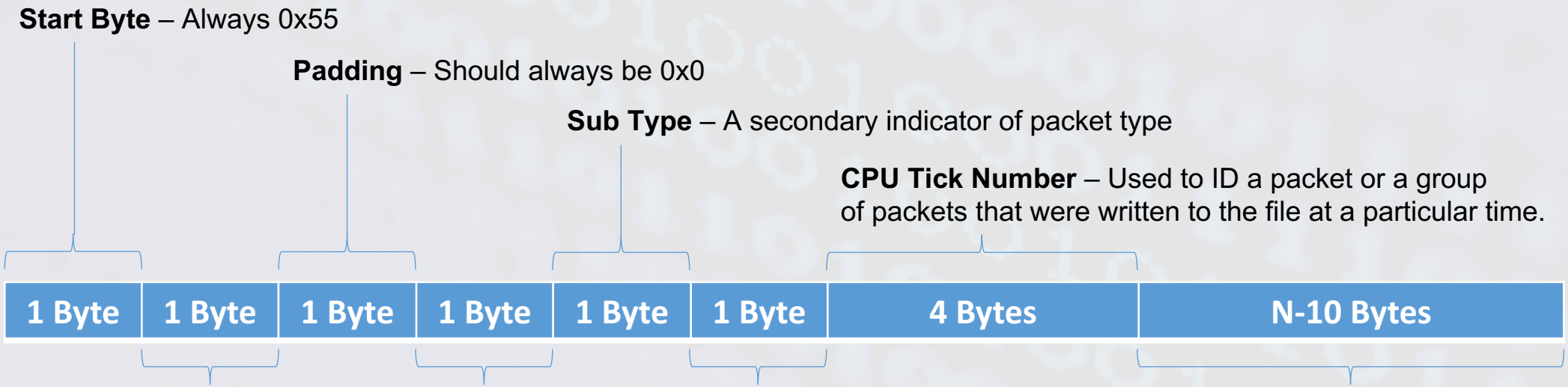
- Binary TXT files are stored on the mobile phone or tablet used to control the drone. TXT flight records can be found at:
  - `Internal storage\DJI\dji.pilot\FlightRecord\`
- Binary DAT files are saved to an internal SD card inside the body of the drone
  - The drone can be connected to the computer, however this is not a forensically sound method of acquisition
  - Disassembly required

Analysis: DAT File

- Logs GPS, IMU, Battery, Remote Control, Flight Status, Motor Data, and more
- A new DAT file is created each time the drone is turned on
- Reverse engineered the DatCon application for file structure



Analysis: DAT File Data Packet Structure



Start Byte – Always 0x55

Padding – Should always be 0x0

Sub Type – A secondary indicator of packet type

CPU Tick Number – Used to ID a packet or a group of packets that were written to the file at a particular time.

Length (N) – the length of the packet in bytes. This includes the start byte to the last byte of the payload

Type – An indicator of the type of packet (GPS, Battery, Remote Control, etc.)

Msg – Usually 0x0, but can be other values depending on the type and sub type.

Payload – Payload is N-10 bytes in length, where N is the length defined in the header. Payload is Encrypted.

Parsing & decryption algorithms

Algorithm 1 DAT packet parsing algorithm

```
1:  procedure DROP(inputFile, outputFile)
2:      meta ← inputFile metadata
3:      in_file ← open inputFile in read binary mode
4:      file_header ← read bytes 0 to 127 of in_file
5:      build ← unpack file_header[16:21] to string
6:      if build != "BUILD" then
7:          exit()
8:      out_file ← open outputFile in write mode
9:
10:     byte ← read next byte from in_file
11:     message ← new Message()
12:     while byte length != 0 do
13:         if byte != 0x55 then
14:             byte ← read next byte from in_file
15:         else
16:             packetLength ← read next byte from in_file
17:             padding ← read next byte from in_file
18:             if padding == 0x00  &  packetLength >0 then
19:                 header ← read next 7 bytes from in_file
20:                 payload ← read packetLength - 10 bytes from in_file
21:                 thisPacketTickNo ← unpack header[3:7] to integer
22:                 if message.tickNo == NULL then
23:                     message.tickNo ← thisPacketTickNo
24:                 if thisPacketTickNo != message.tickNo then
25:                     if message.addedData == TRUE then
26:                         out_file ← message.getRow()
27:                         message.addedData ← FALSE
28:                 message.addPacket(packetLength, header, payload)
29:                 byte ← read next byte from in_file
30:             else
31:                 byte ← padding
32:     out_file ← message.getRow()
33:     in_file.close()
34:     out_file.close()
35:     return
```

1. Input file is opened to determine if it is a DJI DAT format by extracting 128 byte header and then checking bytes 16-20 for "BUILD". If found, program continues, otherwise, it stops
2. If file is a DAT file, next byte (128) is read, and message instance is instantiated
3. Program enters a loop to end of file extracting packet length, header, and payload for each packet and updating message. Note: When data from a new payload is added to message, it must be first decrypted using Algorithm 2
4. We write a row to the file each time we see a tick number change

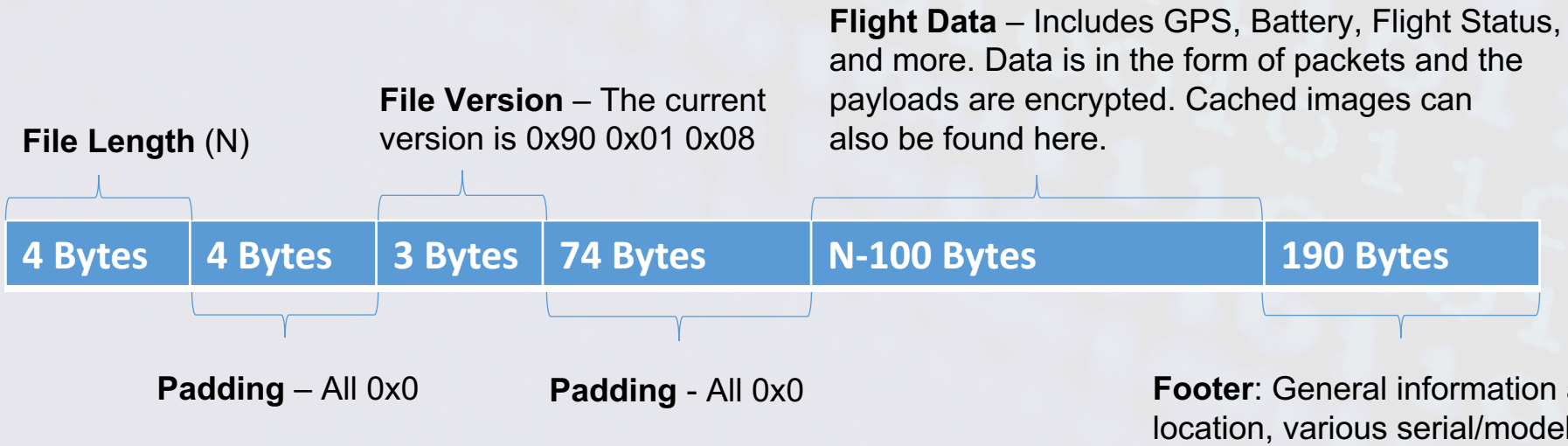
- *Key is simply the tick number for the packet modulus 256*
- *Each byte in the payload is XOR'd with the key*
- *A very weak encryption algorithm*

Algorithm 2 DAT payload decrypt algorithm

```
1: procedure DECRYPT(payload, tickNo)
2:     xorKey ← tickNo MOD 256
3:     decryptPld ← []
4:     for byte in payload do
5:         append (byte XOR xorKey) to decryptPld
       return decryptPld
```

Analysis: TXT Flight Data

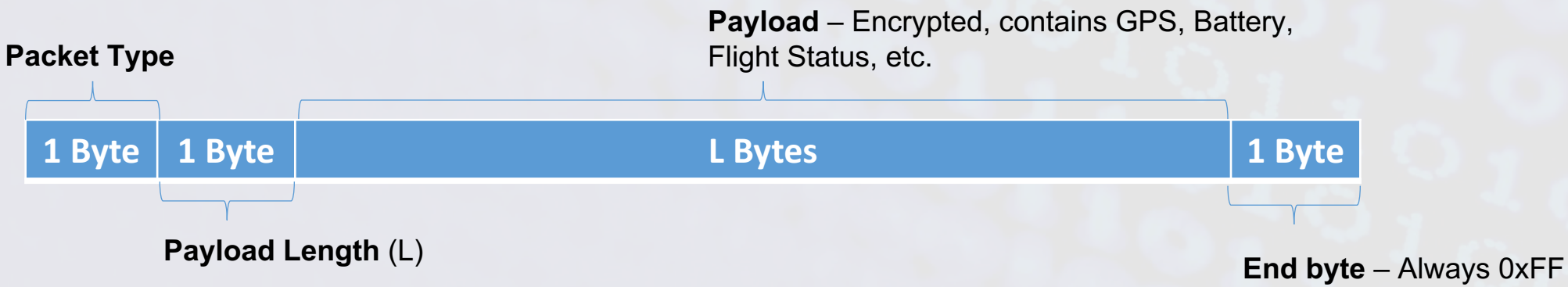
- Stored on the mobile device
- Logs similar types of data as the DAT files
- Uses a stronger encryption scheme
- TXT File structure:



Analysis: TXT Flight Data

UNHcFREG

- What do we know about the packet structure?
  - Comes in a variety of lengths based on type
  - Encrypted using a much stronger method then the DAT files
  - Always end with 0xFF



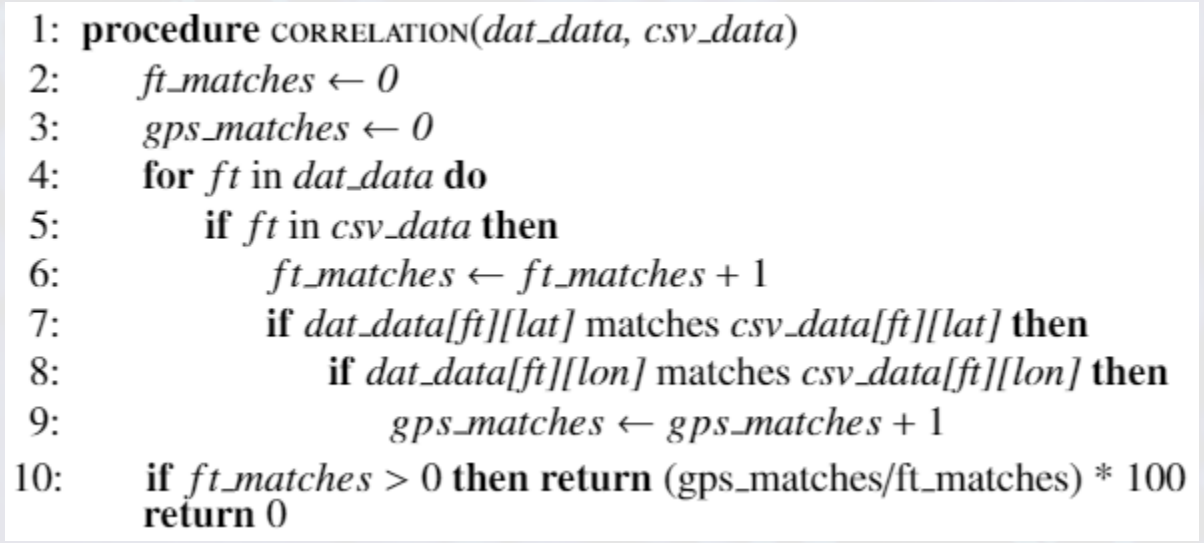
Analysis: TXT Flight Data

- We currently do not have a method for decrypting the flight data payloads
- Attempted to reverse engineer DJI GO application
  - Found libFRencrypt.so – proprietary library for encryption and decryption of the TXT files
  - Extracted C and Assembly code using IDA Pro but have yet to figure out the input parameters
- HealthyDrones **was** the only resource that can be used to extract flight data https://healthydrones.com/
  - DatCon has incorporated TXT parsing into their program

Analysis: DAT to TXT Correlation

- Dates and times for each event can be extracted from the TXT file
  - Currently not possible for DAT files
- DAT files can be matched with TXT files
- This can be done by matching GPS coordinates for given flight times for both files
- But why bother?
  - Determine the dates and times of events for each DAT file
  - Provides further proof that a mobile device is linked to a drone

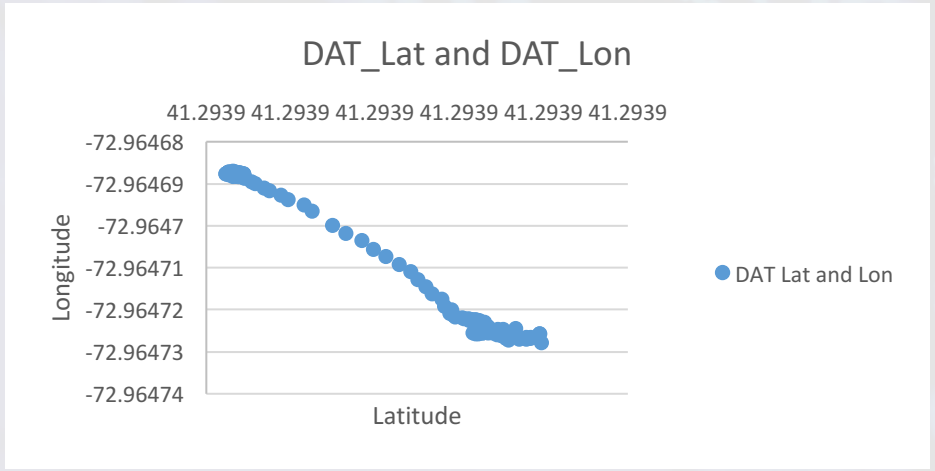
Analysis: DAT to TXT file correlation

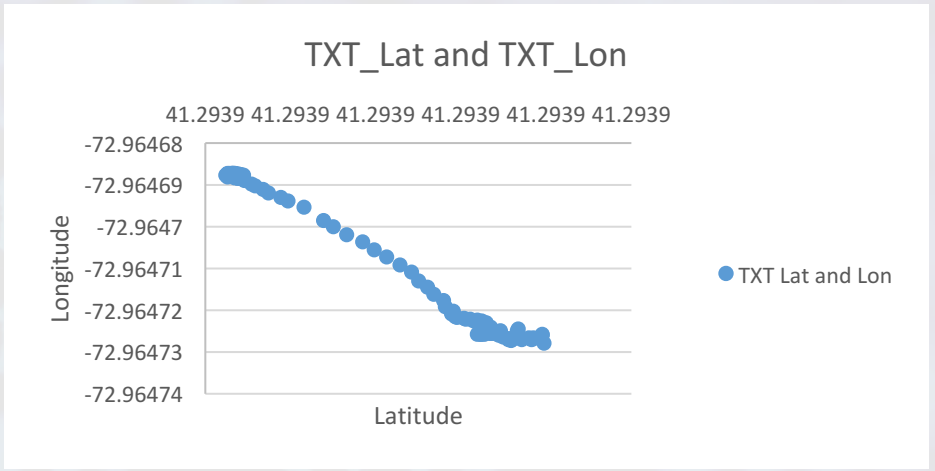


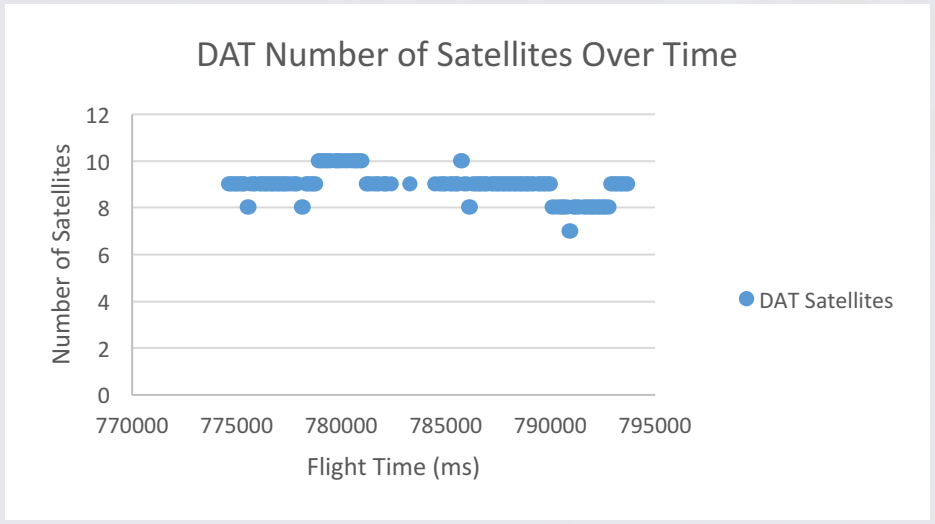
```
1:  procedure CORRELATION(dat_data, csv_data)
2:      ft_matches ← 0
3:      gps_matches ← 0
4:      for ft in dat_data do
5:          if ft in csv_data then
6:              ft_matches ← ft_matches + 1
7:              if dat_data[ft][lat] matches csv_data[ft][lat] then
8:                  if dat_data[ft][lon] matches csv_data[ft][lon] then
9:                      gps_matches ← gps_matches + 1
10:     if ft_matches > 0 then return (gps_matches/ft_matches) * 100
        return 0
```

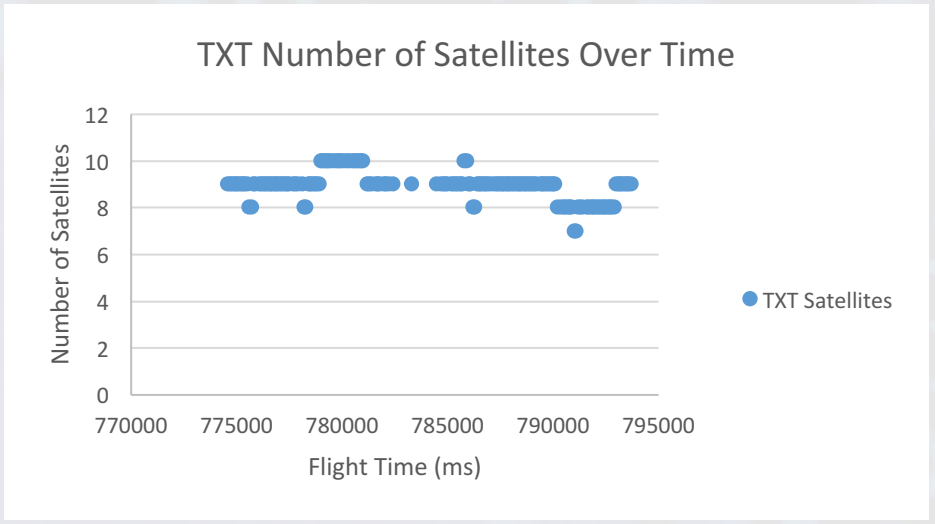
- Cycles through data in DAT file and TXT csv files
- First checks to see if there is a flight time match
- Then checks if there is a match on latitude and longitude for a given flight time match
- Returns percent confidence of DAT to TXT file correlation

Results: DAT to TXT Correlation

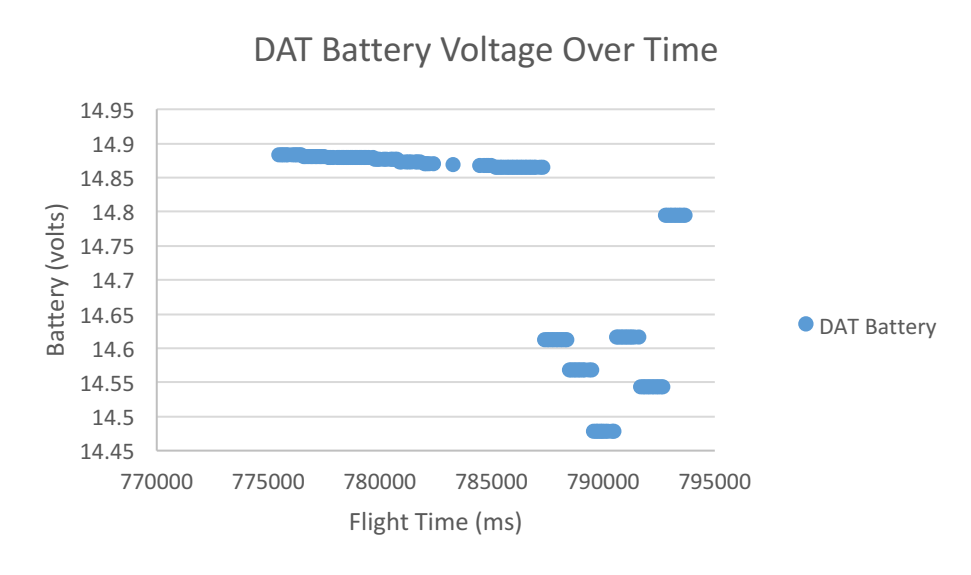


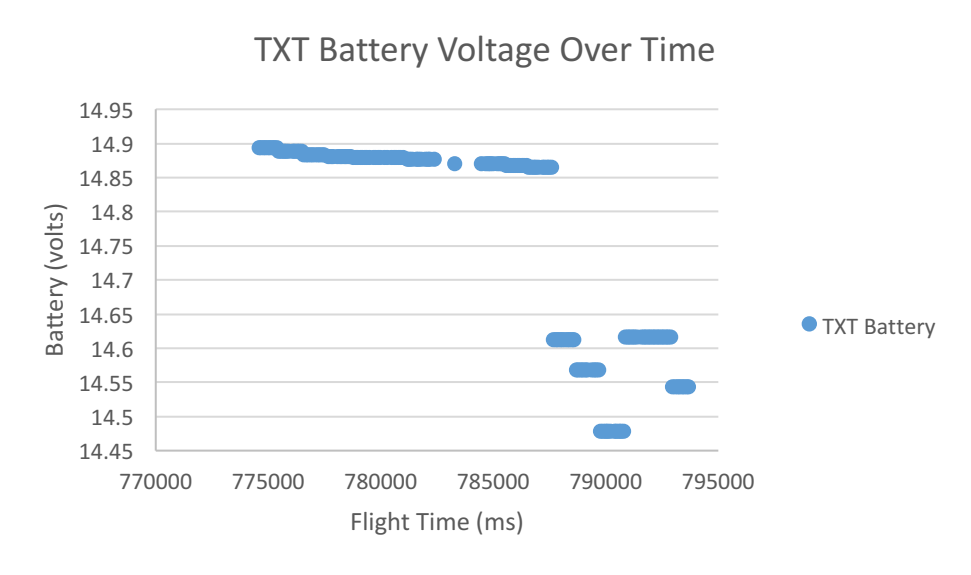






Results: DAT to TXT Correlation





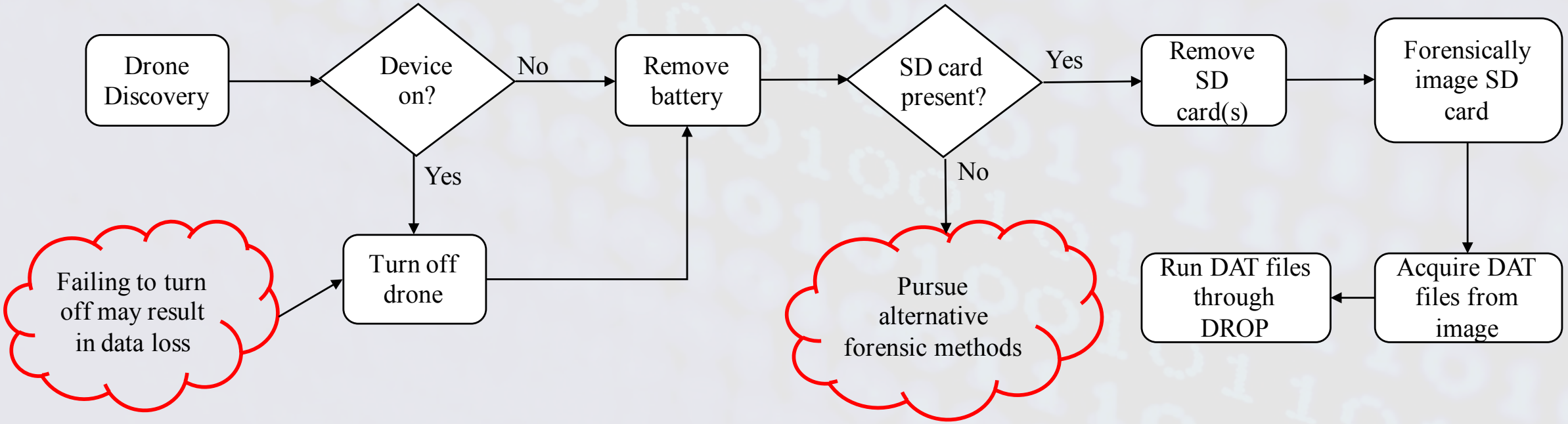
DAT to TXT file correlation standard deviations

Metric	Standard Deviation
Latitude & Longitude	4.99×10^{-6}
Number of satellites	5.55×10^{-1}
Battery voltage	8.49×10^{-2}

Results: Additional Artifacts

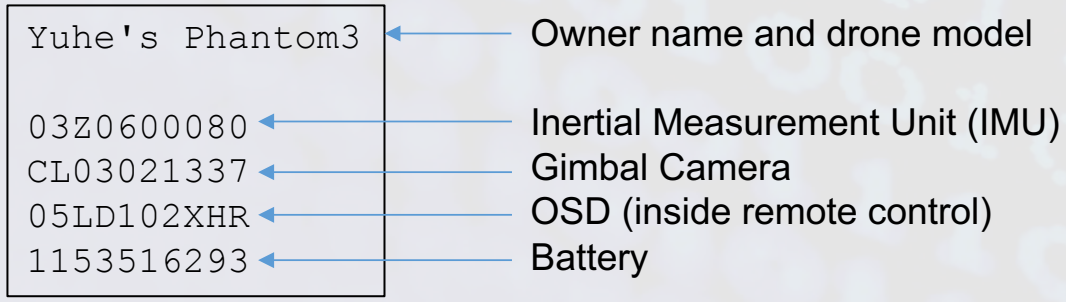
- Images and image EXIF data found on the Gimbal camera SD card and DJI GO cached images: `Internal storage\DJI\dji.pilot\CACHE_IMAGE`
- `dji.db` – A database with information about the app use and history. Data includes:
  - Email addresses
  - Location of last known home point
- `dji.pilot.xml` – Contains information about the last flight point and more
- `flattened-data` – not directly associated with DJI but can be used to obtain the drone's SSID and password

Results: Suggested Investigator Flow



Results: Other Interesting Findings

- The drone will fly without an internal SD card
- An Android device can be correlated to a physical drone and the remote given either a DJI GO TXT file or dji.db



Results: Testing and validation

- We have created several sets of simulated corrupt data in to examine the forensic soundness and reliability of the tool
- Our results also show that DatCon only outputs 1 in 5 records from the DAT files, illustrating that DROP is more comprehensive in its parsing

Future Work

- Continue work reverse engineering DJI GO App and flight record decryption library
- Reverse engineer the firmware
- Try to gain access to drone via SSH or Telnet
- Attempt to hack no-fly zones
  - Create custom hardware to spoof the GPS coordinates of the drone
- Attempt to pull data from newer DJI drone models

Final words

- Results from this work has been used and cited by the Home Office Science - Centre for Applied Science and Technology, United Kingdom in their effort to create and assess drone forensics tools
- This work has results in training intelligence operatives through the United Nations Office on Drugs and Crime (UNDOC) on the forensic analysis of Drones for anti-terrorism purposes (specifically in the Middle East)

Questions?

Download DROP and DJI Phantom III data sets:

Clone DROP on GitHub: https://github.com/unhcfreg/DROP

Link to tool and demo data download: http://tinyurl.com/zj49mq9