DKE LABS

grim-trigger-docs Documentation

Release 0.0.1

Matt Oldfield

Nov 28, 2017

1 Operation Manual **3**

2 Components **5**

3 Maintaining These Docs **7**
3.1 Operation Manual 7
3.2 DDrone Hardware 8
3.3 IOIO (YoYo) documentation 8
3.4 Raspberry Pi 3 8
3.5 Power hub devices 8
3.6 Summary 9
3.7 DDrone Android Permissions 11
3.8 Raspberry Pi 3 12
3.9 Writing documentation 12
3.10 Main Section 13

Info on the general development process of DDrone

CHAPTER

ONE

OPERATION MANUAL

- operation-sec

CHAPTER

TWO

COMPONENTS

- *Hardware*
- *Android App*
- *Raspberry Pi App*

CHAPTER

THREE

MAINTAINING THESE DOCS

- *Writing These Docs*

Operation Manual

Live Senario

1. Turn on the Phantom 3 Standard controller and paired Phantom 3 Standard drone.
2. Connect to the Phantom 3 Standard contoller access point with a phone running the DJI Go (before Phantom 4) app
3. Plug in all three wires on the side labled *power* to the battery at the same time to boot up the pi, give power to the Raspberry Pi 3, Alfa and IOIO board.
4. Wait about 1 minute for it to boot.
5. With an Android device running the DDrone app, plug in the micro USB to the phone coming out of the side labeled *phone*.
6. Accept the permissions as they come in, *Bluetooth*, *pictures and videos*, etc.
7. Once the app is started up and the status says (ready) at the bottom, click `start monitoring` to allow the drone detector to send the signal when a drone has been detected.
8. There is a switch on the opposite side of the phone cable to simulate the detection of the drone. Switch it and a notification should appear saying that a drone has been detected.

#. At any point in the process from here on out there is a button labeled `start jamming` that is a radio frequency jammer failsafe mechanism that will jam anything around the 2.4 GHz portion of the spectrum to disrupt the |**drone**|'s communication. #.

Testing

1. Plug in all three wires to the Raspberry Pi 3 battery at the same time.

DDrone Hardware

Devices Used

Table 3.1: Hardware Devices

Device	Description
Android Device	Hosts the DDRONE application. Must support OTG mode, as all newer devices do, for proper IOIO board interfacing.
Alfa AWUS036H Wireless Network Adapter	Enables monitor mode in order to deauthenticate the pilot and crack the drone password.
IOIO-OTG Board (v2.2)	Used to interface the Android device with both the jammer and drone detector via digital I/O pins.
Raspberry Pi 3	Hosts the Alfa (as monitor mode is required) and cracking utilities.
RAVPower 20100mAh USB Type-C Power Bank	Used to power the Raspberry Pi, IOIO board and to provide additional power to the Alfa.
GrimBox Enclosure	A 3D-printed enclosure custom designed to house the aforementioned hardware. STL is available in the Github repository.

IOIO (YoYo) documentation

IOIO usage info

- IOIO-OTG V2.2 on Sparkfun

Raspberry Pi 3

Raspberry Pi 3

- Specs

Power hub devices

Rav Power Portable Charger

- Rav Power Site

Summary

Configuration

Summary

DDrone is an Android application that, when coupled with the necessary hardware, can detect, crack, hijack, and/or RF jam hostile DJI Phantom 3 Standard drones. It can deauthenticate the pilot of an enemy drone, crack the drone's password, and then connect to the drone and force a Return to Home or Land Immediately command. Deauthenticating the pilot destroys his video feed. Forcing the drone to land destroys the threat. DDrone has several core components:

DDrone Android Application

The DDrone Android application itself. Requires an Android device with OTG support for IOIO-OTG board integration.

Raspberry Pi 3 and Alfa USB Network Adapter

The Alfa wireless adapter is capable of monitor mode, which is required to deauthenticate and ultimately crack the enemy drone based on standard DJI mac addresses.

The Raspberry Pi houses the ddrone-pi scripts, which receive commands from the Android application and perform password cracking. The cracked password is then transmitted to the Android application via bluetooth.

After receiving the password, the Android device will automatically connect to the drone while the Raspberry Pi continues to send deauth packets to the pilot, destroying its video feed and ability to send commands through the DJI GO app. The pilot must then resort to manual LoS flight.

Once connected to the drone, the Return to Home or Land Immediately command can be sent. By expanding the DDrone Android application (in the future), intel can easily be gathered – such as retrieving takeoff GPS coordinates, contents of SD cards, MAC address, and other logs that contain information that can ultimately aid in identification of the pilot.

Drone Detecting and Jamming

The DDrone Android application is designed to work with the IOIO-OTG board for integration with external hardware. This includes a digital input (for integrating with a drone detector) and a digital output (for integrating with an RF jammer), along with UI components to activate each action.

If a drone is detected, the DDrone Android application will alert the user using the highest possible alert level in the Android ecosystem (audio, vibration, LEDs if phone-equipped, etc.) The application can then begin scanning and takeover activities, as directed by the user.

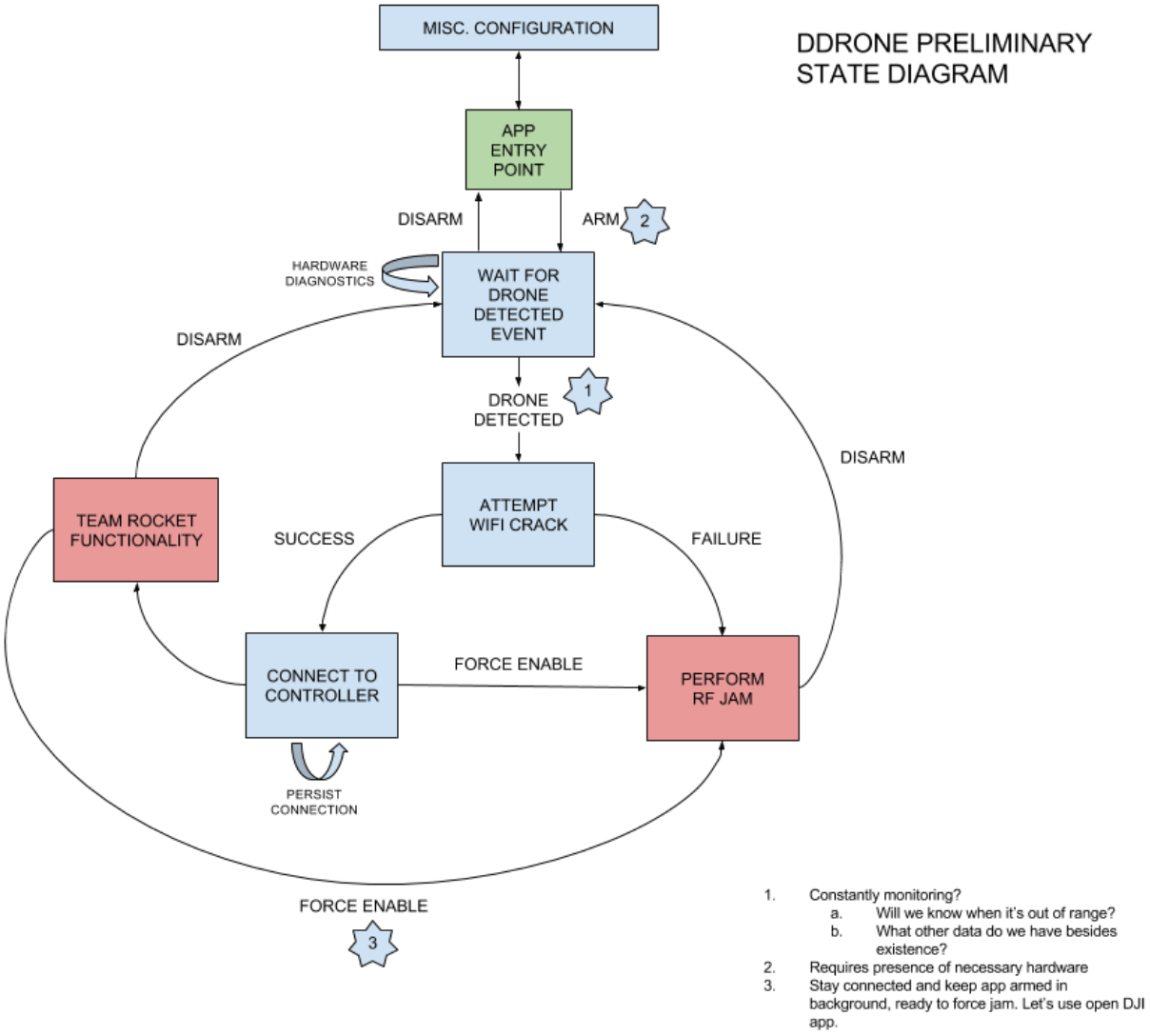
This behavior can be simulated by flipping a switch connected to the IOIO.

Should a takeover or crack fail, the user can request a jam and the drone jammer circuitry (designed/provided by SOFWERX) will be activated.

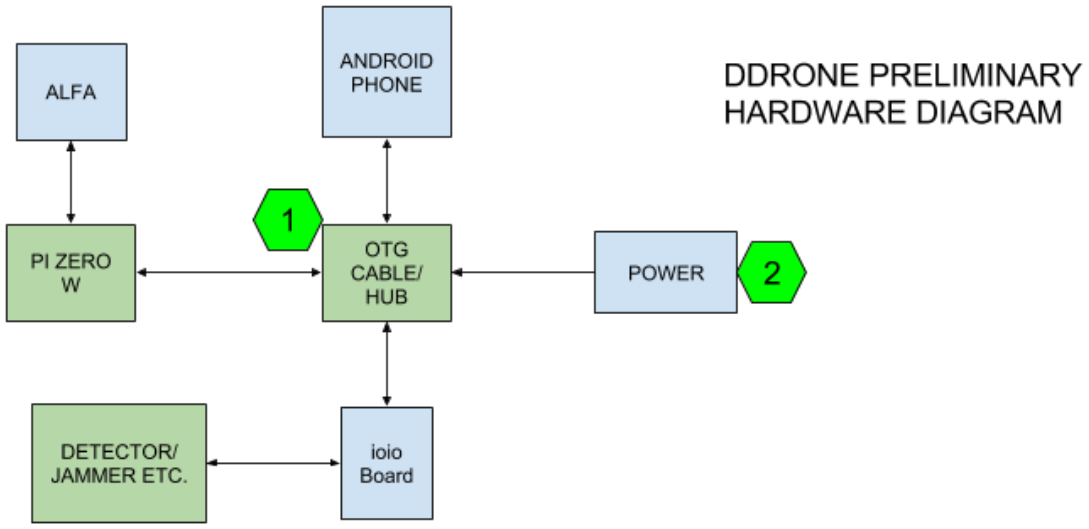
This behavior can be simulated by illuminating an LED connected to the IOIO.

State Diagram

State is managed using the `SharedPreferences` library.



Hardware Diagram



DDrone Android Permissions

Permissions are declared in `AndroidManifest.xml`.

Table 3.2: Android Required Permissions

Permission	Needed For
WRITE_EXTERNAL_STORAGE	Required for the **SharedPermissions** lib
READ_EXTERNAL_STORAGE	Required for the **SharedPermissions** lib
ACCESS_NETWORK_STATE	Required for general functionality.
MOUNT_UNMOUNT_FILESYSTEMS	Required for general functionality.
CHANGE_WIFI_STATE	Connect with the drone's controller once password has been cracked.
INTERNET	Authenticate once with DJI SDK.
BLUETOOTH	Used for communicating with the Raspberry Pi to transmit cracked passwords.
BLUETOOTH_ADMIN	Used for communicating with the Raspberry Pi to transmit cracked passwords.
ACCESS_COARSE_LOCATION	Used for interacting with a hijacked (cracked) drone via the DJI SDK.
ACCESS_FINE_LOCATION	Used for interacting with a hijacked (cracked) drone via the DJI SDK.
VIBRATE	Used for DRONE DETECTED DDRONE alerts.
WAKE_LOCK	Used for DRONE DETECTED DDRONE alerts.
SYSTEM_ALERT_WINDOW	Used for DRONE DETECTED DDRONE alerts.
READ_PHONE_STATE	Used for DRONE DETECTED DDRONE alerts.

Raspberry Pi 3

Bluetooth

Install on pi

```
$ sudo apt-get install bluez python-bluez
```

Append `DisablePlugins = pnat` to `/etc/bluetooth/main.conf` because the `pnat` plugin messes with bluez

Make device discoverable

```
sudo hciconfig hci0 piscan
```

Set the device name

```
sudo hciconfig hci0 name 'Device Name'
```

Wake console from sleep

Useful for testing with the pi commands and there is an hdmi screen plugged in.

```
sudo bash -c 'echo -ne "\033[9;0]" > /dev/tty1'
```

Writing documentation

Basic Structure

```
Main Section
============

.. Comment (include file below)
.. include:: ../vars.rst

.. Link to secondary section that can be referenced anywhere
.. _secondary-sec:

Secondary Section
-----------------

Some regular text.

Bullet List

        * Bullet 1
        * Bullet 2
        * Bullet 3

Tertiary Section
~~~~~~~~~~~~~~~~

Some tertiary text.
```

```
.. Include any file as the literal text
.. literalinclude:: basic-structure.rst
```

Writing Sphinx and reStructured text documentation

- *Curated list of Sphinx docs <awesomeSphinxDocs_>_* - A curated list of awesome extra libraries, software and resources for Sphinx (Python Documentation Generator). Inspired by awesome-sqlalchemy.
- reStructured text for writers - Blog post about reStructured Text
- reStructured text documentation - Official documentation
- reStructured text cheat sheet - Cheat sheet written by matplotlib creators
- Sphinx documentation - Official documentation
- Read The Docs Sphinx - RTD styleguide

Main Section

Secondary Section

Some regular text.

Bullet List

- Bullet 1
- Bullet 2
- Bullet 3

Tertiary Section

Some tertiary text.

```
Main Section
============

.. Comment (include file below)
.. include:: ../vars.rst

.. Link to secondary section that can be referenced anywhere
.. _secondary-sec:

Secondary Section
-----------------

Some regular text.

Bullet List

        * Bullet 1
        * Bullet 2
        * Bullet 3
```

```
Tertiary Section
~~~~~~~~~~~~~~~~~~

Some tertiary text.

.. Include any file as the literal text
.. literalinclude:: basic-structure.rst
```