Perl, Raspberry Pi, IOT and Citizen Science

Motivation:
- Airplane noise over my house
- Contact with HACAN and HACAN East
- Part of IoVMT (the Internet of Very Messy Things)
- Citizen Science, why oh why? (Volkswagen, French Chernobyl cloud etc.)

This will go onto my website later on, with the hyperlinks, so no worries for photos etc.
Hardware for Project
This is a pretty complete decoded ADS-B(roadcast) record, but they vary and the subfield order varies too, see decoding ads-b for raw format. Starts with a *

*8d40615b99449e8ae82c053a7970;
CRC: 000000 (ok)
DF 17: ADS-B message.
   Capability : 5 (Level 2+3+4
   (DF0,4,5,11,20,21,24,code7 - is airborne))
   ICAO Address : 40615b
   Altitude : 7250 feet
   Extended Squitter Type: 19
   Extended Squitter Sub : 1
   Extended Squitter Name: Airborne Velocity
      EW status : Valid
      EW velocity : -157
      NS status : Valid
      NS velocity : -86
      Vertical status : Valid
      Vertical rate src : 0
      Vertical rate : -640

*8d4008795825926141fdfc2253fc;
Fit, the First, Initialise

#use InfluxDB ;  #open source db, specialised for time series
use InfluxDB::HTTP;
use InfluxDB::LineProtocol qw(data2line line2data);  #eek!

# height and decibel cutoffs and defaults, command line can be used
my $cfg = new Config::Simple('/home/pi/aircraft-sensing/bin/read1090.ini');

my $height = $cfg->param("height");
my $db_min = $cfg->param("db_min");
my $test = $cfg->param("test");

# general and associated program paths
my $log_path = $cfg->param("log_path");
my $dump_path = $cfg->param("dump_path");  #path to dump1090 binary
my $test_file_path = $cfg->param("test_file_path");
my $decibels_path = $cfg->param("decibels_path");  #path to decibels.py
Fit, the Second, Pipe in, Parse and Decide (heavily edited to show main features)

open( $fh, '-|', $dump_path );  # pipe in from dump1090
.

while (<$fh>) {  # anytime a transponder record turns up

    # start of transponder record with a *
    # only partial parsing to extract useful fields, other stuff thrown away
    # key : value in the 'useful' subrecords

    If (/^\*/ ) {  

        # ICAO Address   : 40100a
        if ( $key =~ /^ICAO/ ) {  
            $ica = $value;
        }
    }
Fit, the Third, Get altitude and decide whether to write to database

```
# Altitude : 24000 feet
if ( $key =~ /^ Altitude/ ) {
    $value =~ s/\s+feet//;
    $alt = $value;
}

my $noise = get_decibels($decibels_path);  # calls out to decibels.py
# if the plane is lower && noisier than set limits, then store data
if ( $alt <= $height and $noise >= $db_min ) {
    # mysql replaced by influx now, also
    # 'near the ground reading ($height - $alt)
    send_points_to_influx( $log, $ix, $ica, $noise, $result->{tail},
                           $result->{model}, ($height - $alt) );
```
Fit the Fourth, Store (heavily edited to show main features)

sub send_points_to_influx {
    # data2line($metric, $value, $tags_hashref, $nanoseconds);

    my ( $log, $ix, $ica, $noise, $tail, $model, $alt ) = @_; 
    my $influx_line = data2line(
        'overflight',
        { alt => $alt, db => $noise },
        { tail => $tail, model => $model, ica => $ica }
    );

    # didn't want the bother of http agent etc.
    # FIXME: needs to use the configured parameters!
    `curl -i -s -X POST http://localhost:8086/write?db=aircraft --data-binary "$influx_line"`;

    return 0;
}

And around again, until we are bored....
Block diagram

Block Diagram: IoVMT, so far

Aircraft Sensing

Collect from Influx: HTTP REST

Temperature Sensing

Grafana

On demand SMS

Particulates Sensing

Web Access
Grafana Result
Some Takeaways 1

- No point in purity, `dump1090` and `decibel.py` had the 'merit of existing'
- Perl is good at parsing and rich data structures (I haven't decided yet whether to cumulate one overflight and record a maxima, for example)
- IoT is (my opinion) keep it simple, my testing method is basically bathtub, let it run for weeks and see whether and why it fails. Rinse and repeat.
Some Takeaways 2

• *Contingent relation* between the decibels and the overflight, but *when in garden good correlation*

• Preparing tiny computers for unattended operation, anything that fills up storage is turned off, unwanted services off

• Does *systemd* do a good job on monitoring and restarts? Part of current 'research'

• *cron* driven reboots to clear everything out, but how often?
Some Takeaways 3

- Communication always a problem, Wifi (if poss), SMS dongles (if not, temperature sensing),
- Power supply, power and SD card fails
- *Important!* Make this easy for non-tech to implement, SD image or packaging up. Hence config file, as a 'start' towards this.
- It is citizen science, not nerdy-science
- Other current IoVMT projects, particulates, temperature humidity
- Future: protocol and architecture soup see: 
  Summary of Protocols
Thanks!

- @hughbarnard
- hughbarnard.org for the slides sooner or later
- Sorry, don't do facebook, don't do linkedin