Building an RF Scanner Array

(with COTS parts)

Andrew R. Reiter
Researcher, Veracode

First Off

- There is **entirely** too much to cover in one session:
  - Some universities dedicate entire courses to some of the topics covered
- Do not worry if you feel confused
  - be patient -- read more after the talk AND!
- Join the *radiopunks* mailing list (see end of slides)

Increase in RF Traffic

- Reorganization of the Frequency allocation table
- Cost reduction in RF capable devices
- Non-FCC (other) licensed devices that can TX.

Challenges for Us

- What is being transmitted Over-the-Air (OTA)?
- Should what's in the air, *be* in the air?

It is our Duty

- To first better understand:
  - of channel frequency occupancy
  - probability distributions related to channel usage
  - & the expected protocols in different frequency ranges

...All so we can be **aware** of existence of signal...
THEN we work toward understanding signal

Governments are listening, so should WE.

Feel it is my Duty

- To talk about my current array & previous attempts:
  - Failure, success, and goals...
- So others can build their own...
- In order to increase knowledge of OTA use
- **Hopefully** you will get motivated to:
  - build an array and
  - participate in a world wide RF sense network!

Innocent Motivation: I live in the Woods…

…and found this on a hike near my house…

made me wonder what else I couldn't see!

…Drove me to build a scanning array

RF Scanning Array

- A group of RF RXing devices with the purpose of discovering signals in the RF spectrum.
  - Just *power analysis* to reveal noise floor and spikes
- Separate is:
  - demodulation of IQ data to protocols (or up to application layer data)
  - identification of protocols
- **Recognition** of signal **existence** is the 1st step!

Basic Array Components

- Software controllable RF receiving capable device(s)
  - Has ADC producing IQ data
- Antenna(s)
- Cabling/Adapters for Antenna←→RF RXer(s)
- Computer able to
  - control the RF devices and
  - handle IQ data

Component Choice Problem

- Frequency ranges of the RXing device
- Antennas work best for specific ranges
- Multiple RXing devices/antenna vs many antenna
- Storing data? Prepare for lots of it or lose history
- VMs not an option for this: I/O hell.

Yes, there's a lot to consider

- Start small.... with a frequency range that can easily build/buy hardware for
  - Target VHF (30-300M) & lower ends of the UHF (300M-3G) range:
    - Easier to find cheaper parts
    - Lots of fun stuff to "see" in this range:
      - POCSAG, ADS-B, etc
- ...Then expand your desired range support.

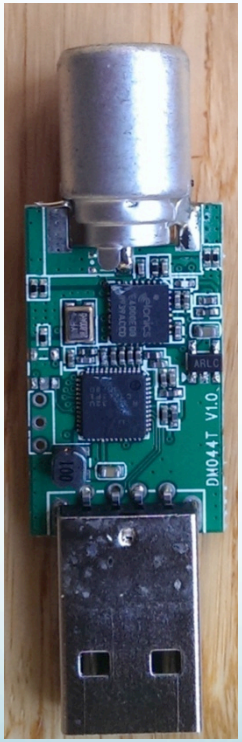
RF Receiving Hardware

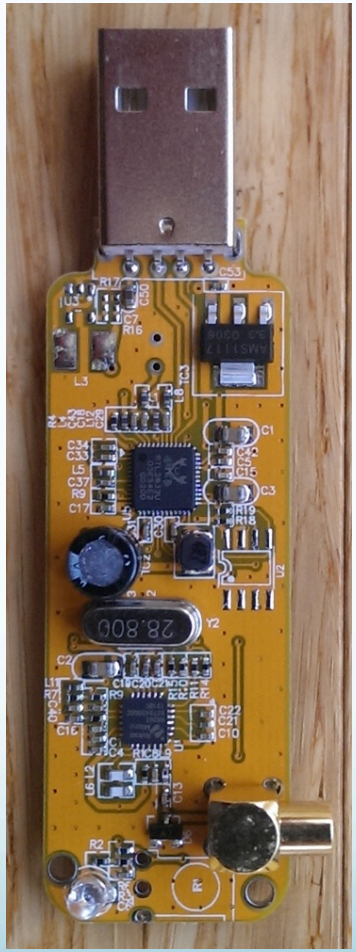
So you know the basis of what I used.

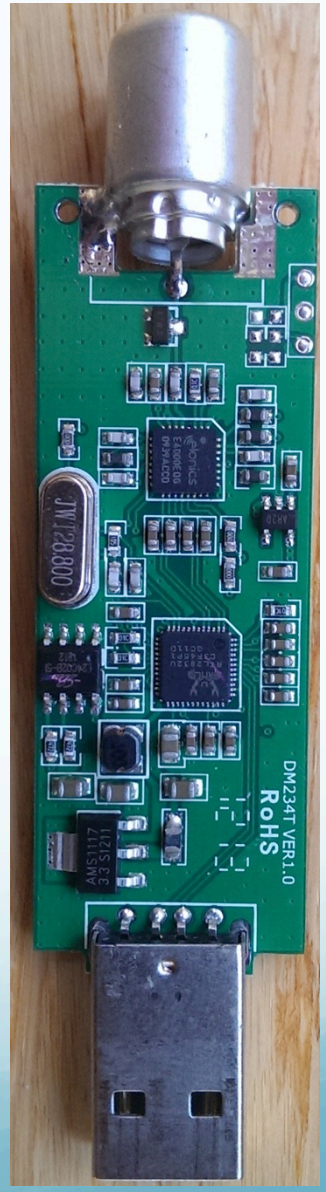
RF Rxers I Used

- Variety of Realtek RTL2832u-based [6] USB sticks
- Tuner chips used ranged from Elonics 4k to Rafael Micro R280D.
  - Depending on the tuner, get different support...
    - At best, looking at 50MHz-2.2GHz with some gaps
- Theory support 3.2M/s, but is well-known that will have data loss ... use 2.4M/s

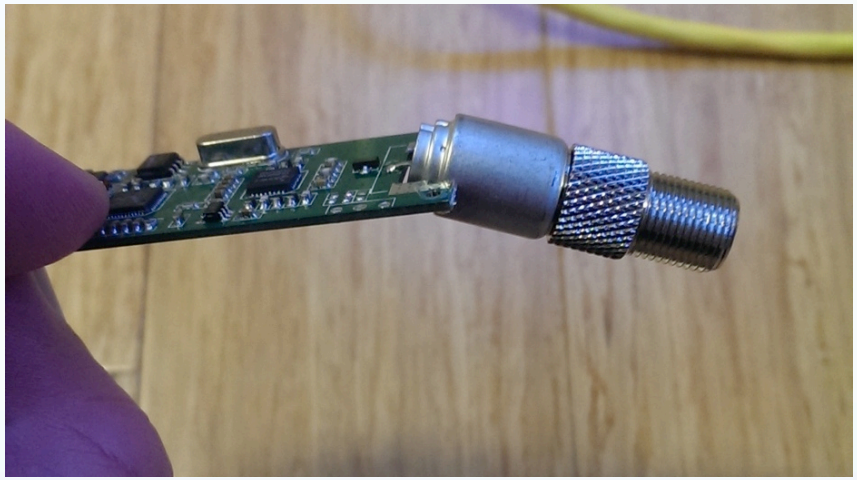
Some examples







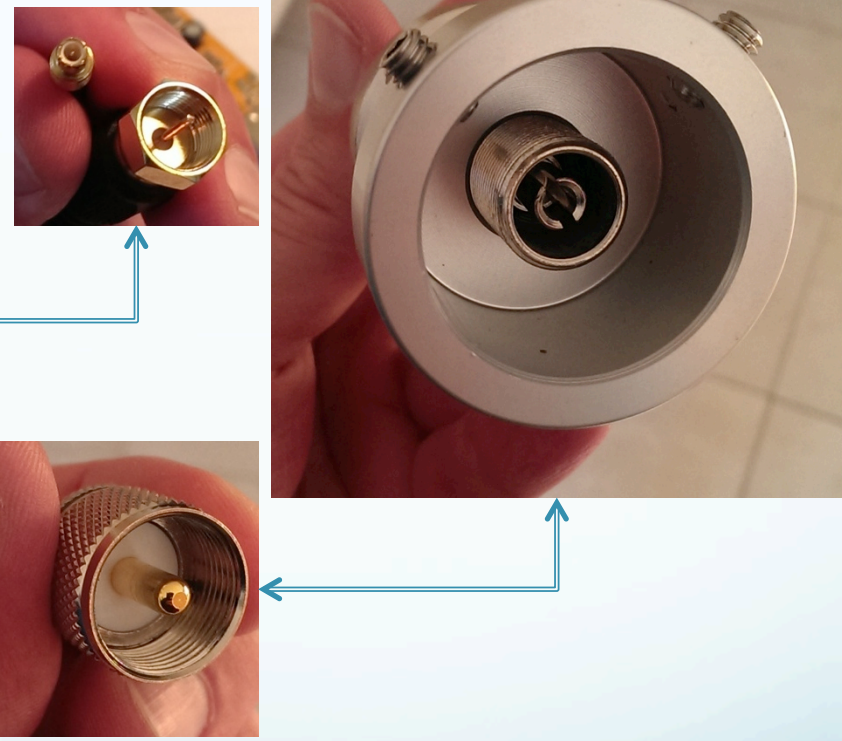
PAL Type Tend to Break



- This is due to weight of cabling..
- Though quite easy to solder back in place...

Cables & Adapters

- Coax (rg6 is fine)
  - Worry about socket quality!
- Adapters:
  - F-type → PAL
  - SMA → MCX
  - PL-259 UHF plugs



All *can* be painful to get when you have all the other hardware… buy a bunch at once so you have extra…

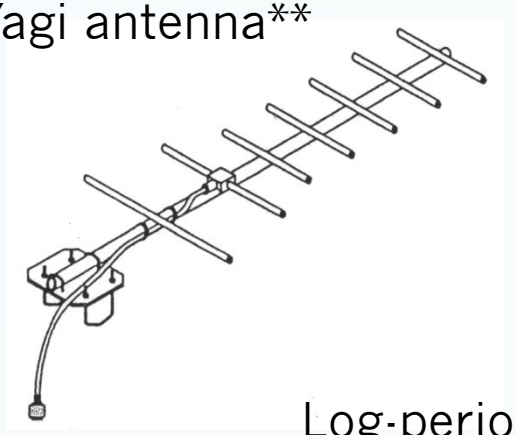
Ordering from eBay usually means ordering from China and 3weeks.

Antennas

- Depends on your:
  - Needs
  - Location
  - What you know about your surrounding area
- Many different types for:
  - Very specific freq ranges vs wide-range
  - (high-gain) directional vs omnidirectional

I have used mostly *discone* omnidirectional but am slowly adding directional to my array

Yagi antenna**



Discone antenna*

Log-periodic***

*http://www.zcg.com.au/Images/base/b51h-discone-antenna.jpg
**http://www.comprodcom.com/data/images/Antenne_de_base/480-70.jpg
***http://en.wikipedia.org/wiki/File:LPDA-Antenna.jpg

Antennas

- Build your own with household items!
  - Some pine wood
  - Screws
  - Coat hangars (stripped of plastic)

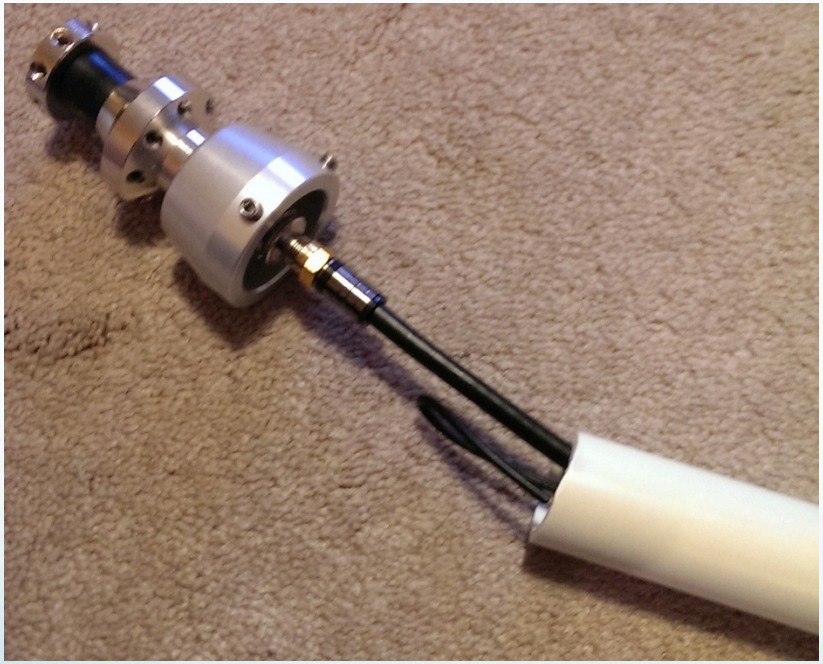
  And you're basically there...

  For an example on a HDTV OTA build:
  http://www.youtube.com/watch?v=7j80C9d1o9Y
  (thanks to the folks at CBC Ottawa for this!)

Antennas & Mounting

- **Mounting height** is crucial [4]:
  - Trees/buildings can block signal!
  - Refraction is testy, so be aware of your surroundings
  - Try to mount on a roof or up in a tree (above other trees)
  - There are *rules of thumb* [5] on height & distance to TXer
- **PVC** is:
  - cheap / can mount most antennas, AND
  - raises the height of your antenna
- **Ground** them: protect you and your equipment
- **Label** them: protect your sanity

PVC Mount for Omni

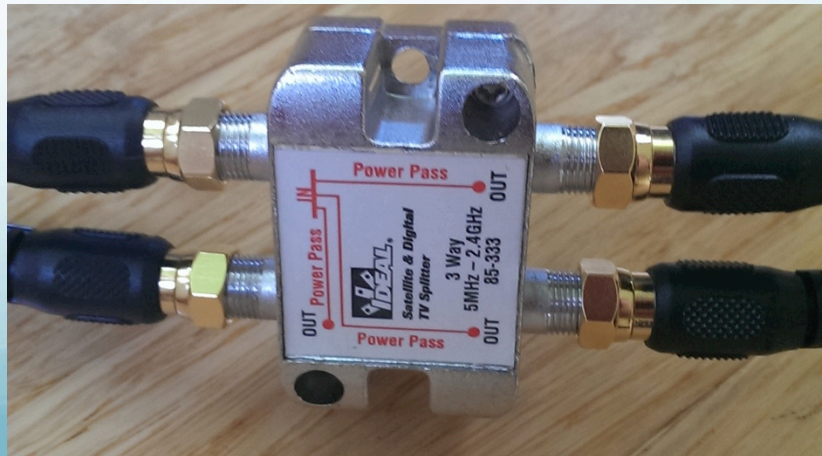


Antennas on PVC on shelf on big hill but with trees :-/

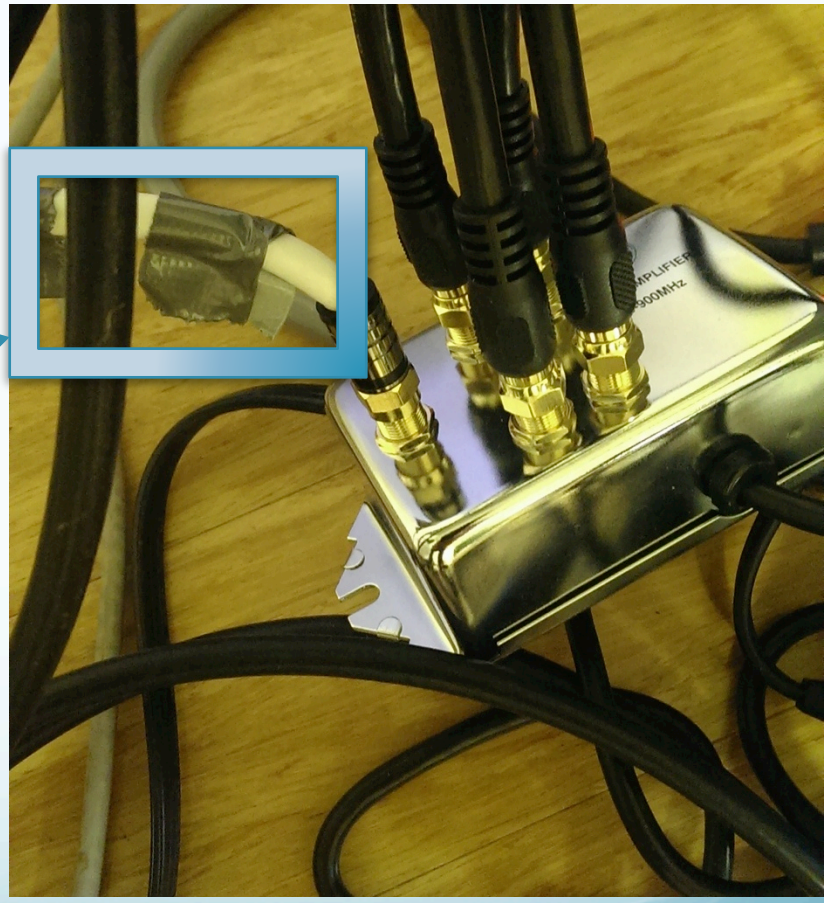
Grounding & Labeling

Antennas & Multiplexing

- Not always easy/possible to mount all → use multiplexor
- Multiplexor blocks split a source signal to N output signals
  - *higher* the *frequency*, *higher* the *cost*
- **Know this trade-off**:
  - *Non-amplified* m-plexors result in dB loss, but
  - *ampflied* m-plexors are noisy … choose your battle.
- I opt to go with amplified.

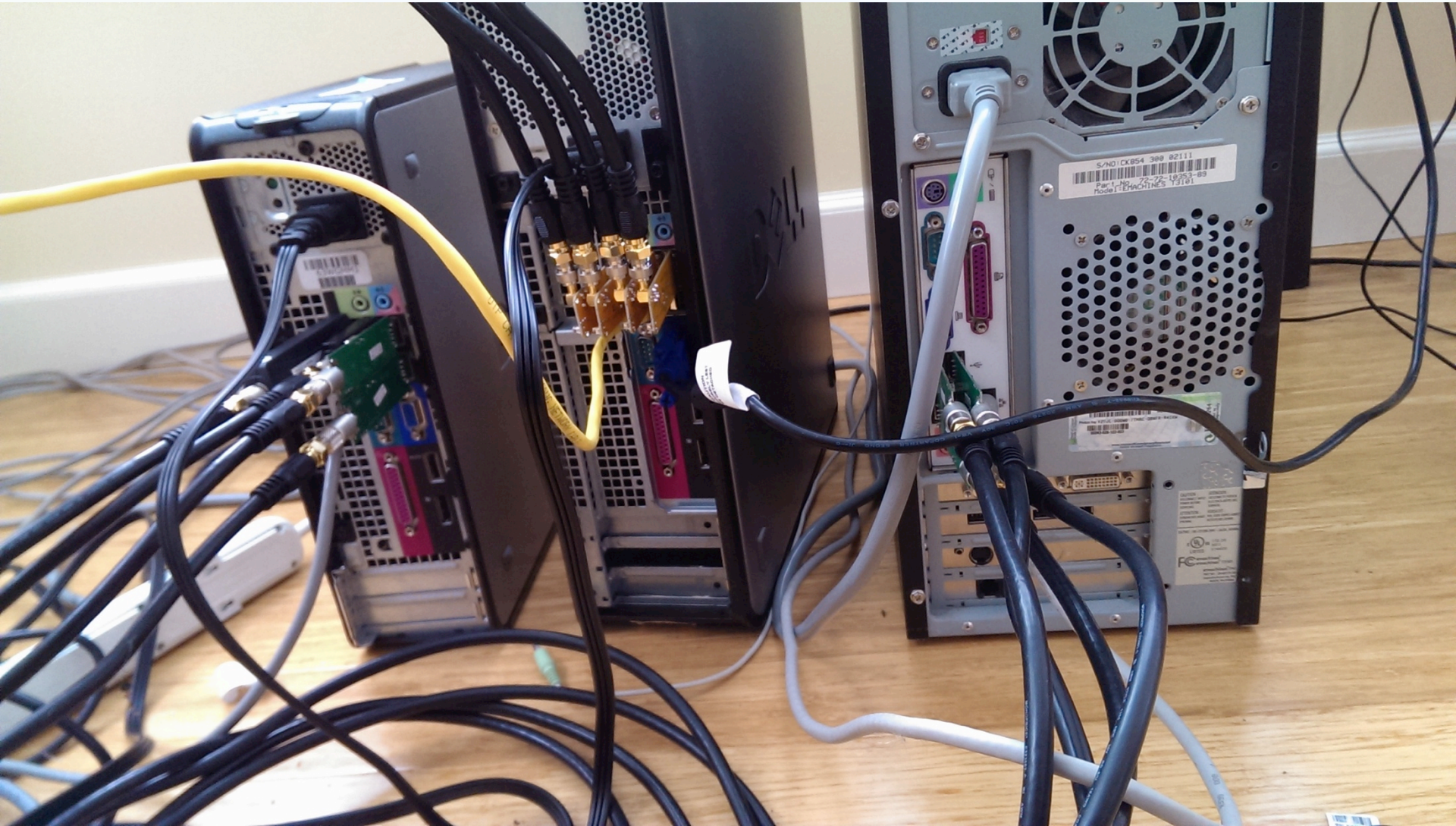


Multiplexor Labeling



How else would I know what goes where? It's a mess of cables!

Finally end at Slave Nodes



Don’t Buy Tetra-USB Cable

- China eBay specials!
- These are a fail
- Using 1 with 4 RTLs caused numerous crashes to occur in *libusb* and *libc*
- Excuse? I wanted Octo-SDR!
- Don’t do this! Save your money!

Software Side

Obligatory Note: it is possible that some other code will do similar work to mine, but when I was starting my work, there was none. For example, GQRX could conceivably support more than one device now, etc ....

Use Distributed Approach

- Lot's going on for one system...
- Positives:
  - Spread out
    - the USB I/O load (the RF capture)
    - CPU intensive FFT
    - Data storage
- Negatives:
  - Networking costs (cable / hardware)
  - Machine costs (power / hardware)
  - Possible complexity increase

Distributed Designs Tried

1. Master-Slave with *pyrtlsdr* [2] driving the USB dongles.
2. Master-Slave with each dongle associated with a *rtl_tcp* process and GNURadio [3] doing workflow
3. Master-Slave with each dongle associated to a C developed code *heavily* based on the Osmocom *librtlsdr* and *rtl_** utilities.

In Each Design

- *Slave nodes are*
  - only machines with RTL-SDRs in them
  - RX RF data and either:
    - Send IQ data to master, or
    - FFT and send power data to master
- *Master* node is
  - primary interaction point for the array
  - intended to control the actions of the RTL slaves.
  - holder of all RX'd data: IQ *or* Power values
  - intended aid in display of resultant data
    - GUI, or
    - Scripts to generate reports

High Level Differences

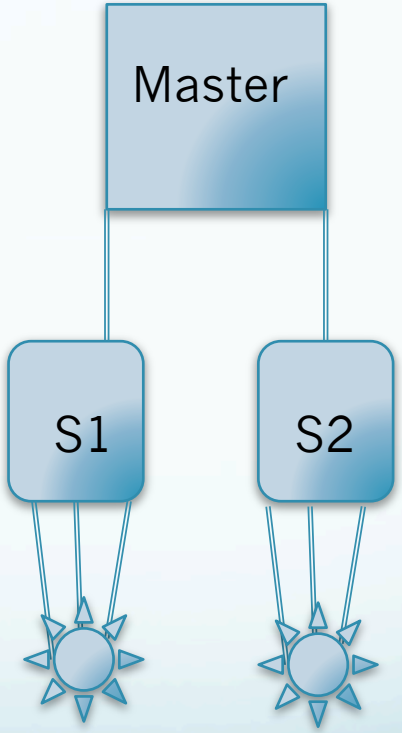
- How data is captured:
  - *PyRTLSDR* vs. *rtl_tcp* vs. *librtlsdr*+other
- Where IQ data is analyzed:
  - All on *master* vs. hybridized *master-slave*
- Functionality:
  - Single purpose scan-only vs. multi-purpose

The first 2 designs listed are discussed less as I settled on the 3rd after implementation of all

1. Master with PyRTLSDR nodes

- Each slave process is given some block of the spectrum from 50-900MHz:
  - Then each is broken to 2M chunks
- Slaves capture IQ data and push to the Master
- Master attempts real time analysis on data to find spiking frequencies:
  - => FFT per capture per slave *on one machine*
  - Then add the network IO & disk IO

The problem with multiple channels hitting one machine raw is the ability to keep up!



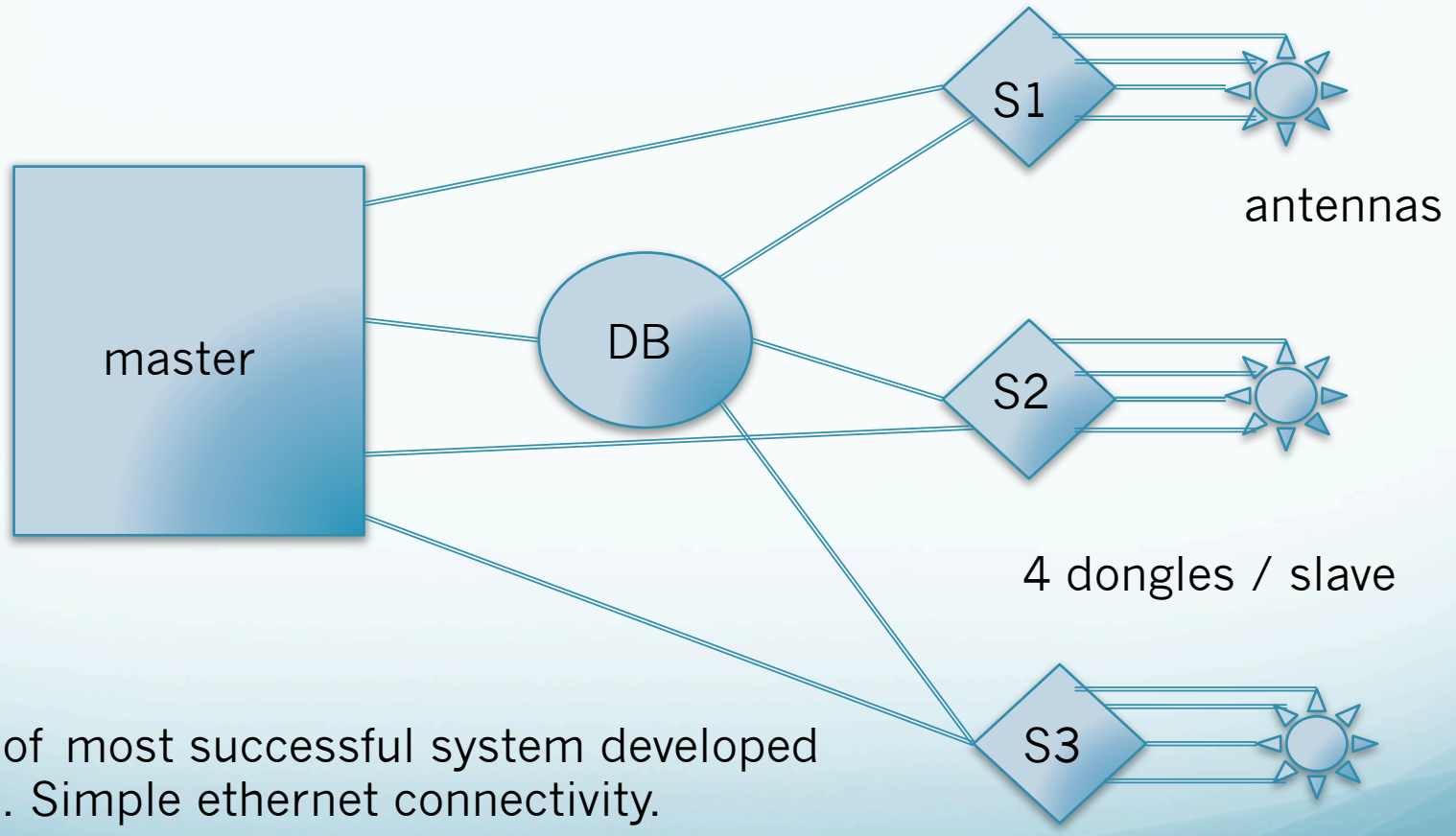
2. Master with rtl_tcp nodes

- Each *slave* runs *rtl_tcp* for each *dongle*
  - Pipes the IQ data over the network
  - Removes Python bindings layer
- *Master* runs python script, generated by GNURadio Companion [3], that uses
  - OsmoSDR *source* for each *rtl_tcp* instance.
  - FFT blocks → frequency domain
  - Custom written GR *sink* block:
    - aggregates the FFT'd data
    - performs logging / basic analysis

rtl_tcp method

- It *will* try to crush your network and your single machine trying to process multiple nodes.
- Periodic crashes from deep within GNU Radio
  - Unsuccessful at debugging this
- Bonus points: get to use all the great blocks that come with GNU Radio!
  - So some flexibility in design

3. Master with custom nodes



Topology of most successful system developed (of the 4). Simple ethernet connectivity.

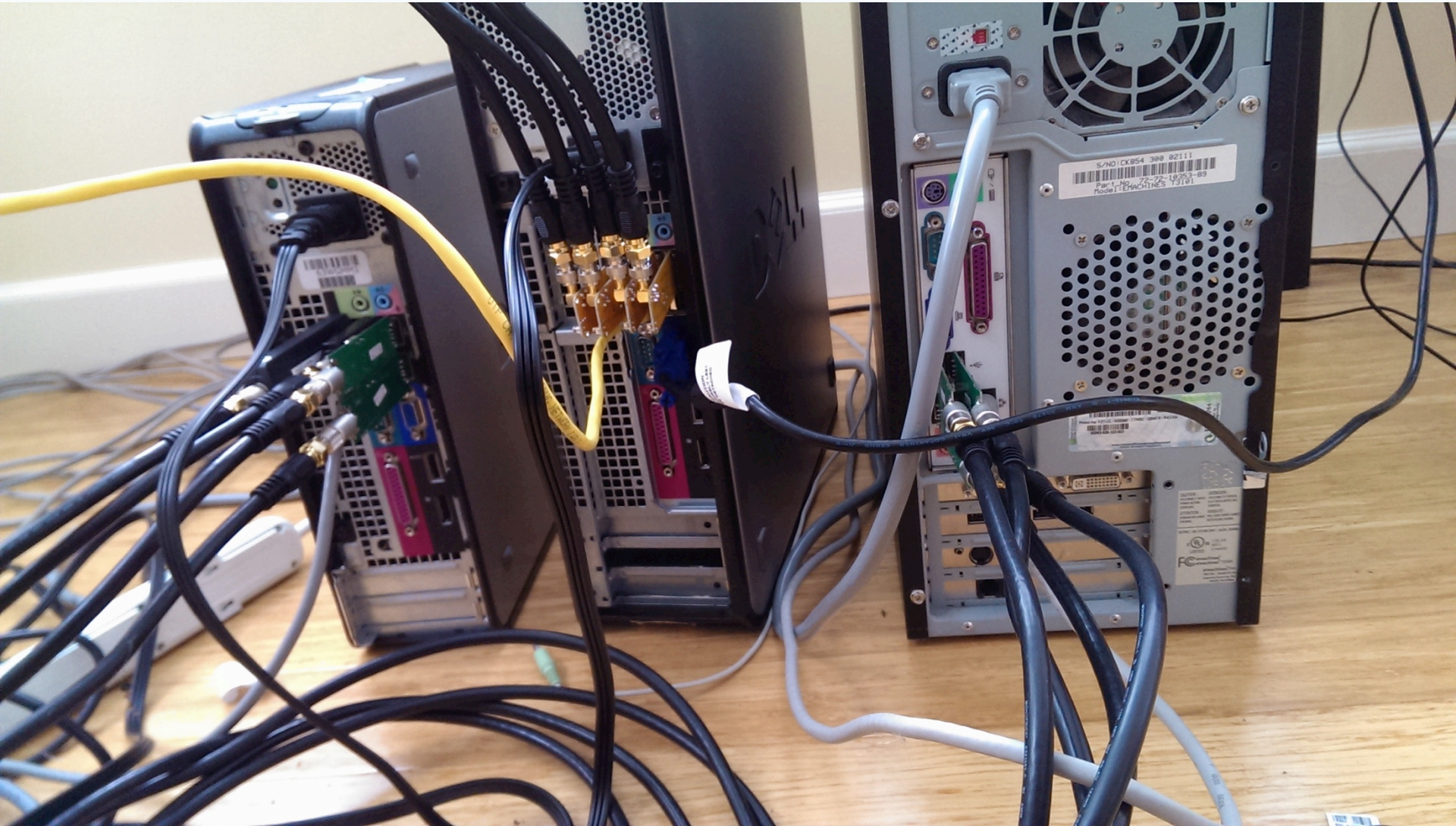
Design Goal

- Push all FFT work to *slaves* to reduce network load
- Increase functionality without taking away from the power analysis:
  - Add raw IQ file capturing
  - Add real time streaming of IQ data via *rtl_tcp*
- Take advantage of
  - *librtlsdr* and *rtl_** tools
  - GNU Radio framework features

Slave Nodes

- Each slave runs instance of *megarrmon* (mega-array-monitor) for each dongle.
- *Slave*'s **primary** goal is to:
  1. Capture RF data from RTL-SDR
  2. Transform data to frequency domain (FFT)
  3. Insert power values into the database
- The *master* can *command* the slave nodes to:
  - Do raw IQ capture to a file & upload to NFS share, &
  - Start *rtl_tcp* on given port.

Slave Nodes (as seen earlier)



Database Node

- The *database* is there for:
  - Node information:
    - Device index
    - antenna capability
    - latitude/longitude
    - Status
  - Frequency block distribution
  - Conduit for command dissemination
    - Stop/start scan
    - Stop/start stream
  - Store power data
- node/block dist config'd via script ahead of time

Master Node

- The *master* is intended to:
  - Control the array via commands into DB:
    - start/stop/exit all or individual nodes
    - Direct node(s) to:
      - do raw captures to file
      - turn into a rtl_tcp streaming node
    - All of the above is done via a GUI
  - Act as NFS server for storage of raw capture files
  - GUI sources data from the *database node*

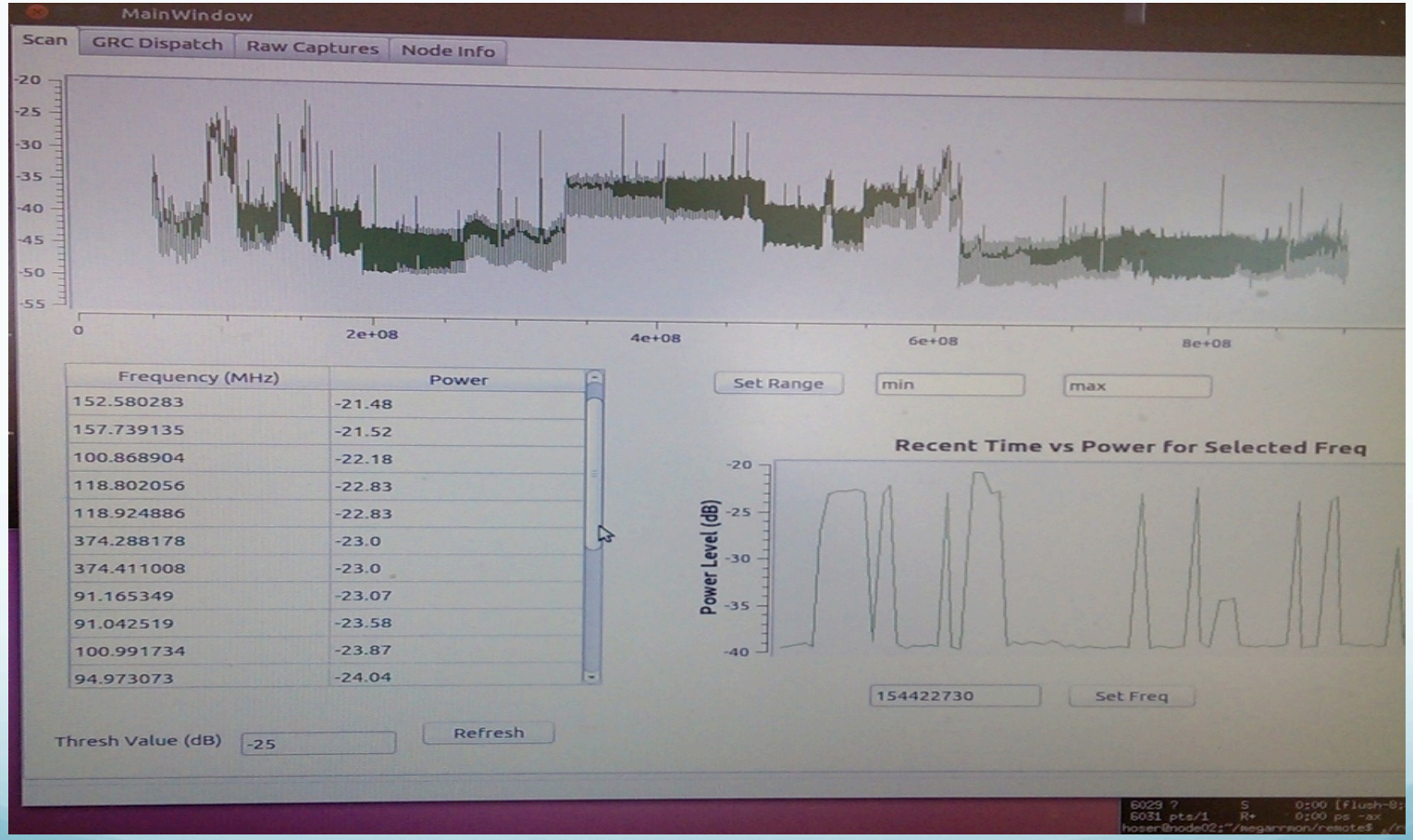
Master GUI

- Power over spectrum graph
  - frequency vs. power (dB)
  - Dynamically configured view range
  - *benefit*: see active parts of spectrum
- Power over time graph
  - Choose a frequency
  - “recent time” vs. power (dB)
  - *benefit*: see the pattern of power for a frequency
- Both graphs refresh with new data…

Master GUI

- Power Thresh Breakers ~~ Strong Signal Analysis
  - Set some dB value above the noise floor &
  - Set last N minutes to search through
  - And generates the list of frequencies breaking that dB barrier
- Launch GNURadio scripts for streaming nodes
  - Currently not much supported except WB FM demod
- Point-n-Click raw IQ capture to file

Master UI



Can you see the 3 different antenna blocks in graph? :-/

My Working Array

- Currently have a 3 slave node (+1 master +1 DB) system
  - Scanning 50MHz to 900MHz
  - 4 RTL-SDRs per node = (12) 2MHz channels
  - 3 omnidirectional UHF/VHF antenna
    - 3 4-way amplified multiplexors
  - I have had it running for >7 days at a time with no memory leaks/crashes; & ran many other times for just few days

Node Hardware / Software

- Master
  - 64 bit with 8GB RAM
  - Big disk for DB
- Slaves
  - Mix of 32 bit and 64 bit machines
  - Not more than 2GB RAM in each
- All running Linux
- Idea is to use machines just laying about!

My Working Array (cont)

- 50-900MHz block is split into 3 chunks & assigned to a slave host
- Each sub-block is broken into more blocks & assigned to each actual RX device on the slave
- Then breaks those chunks down into the 2M bandwidth samples sizes

Freq Breakdown

Example: choose freq & stream

I removed video due to slide size & PDF conversion.
I can upload or refilm one this week (for better quality/smaller size) OR build your array ;-)

Example: change dB thresh, select freq, & stream

I removed video due to slide size & PDF conversion.
I can upload or refilm one this week (for better quality/smaller size) OR build your array ;-)

Minimal Result Data

- Often performance results depend on what the array is doing:
  - Only scanning?
  - Is a node, that usually scans, doing raw IQ captures?
- The system is doing approximately:
  - 1 power sample per frequency / 3 seconds
  - 1GB increase in DB size per hour
  - These are wimpy stats … not concrete.

Next for this Array

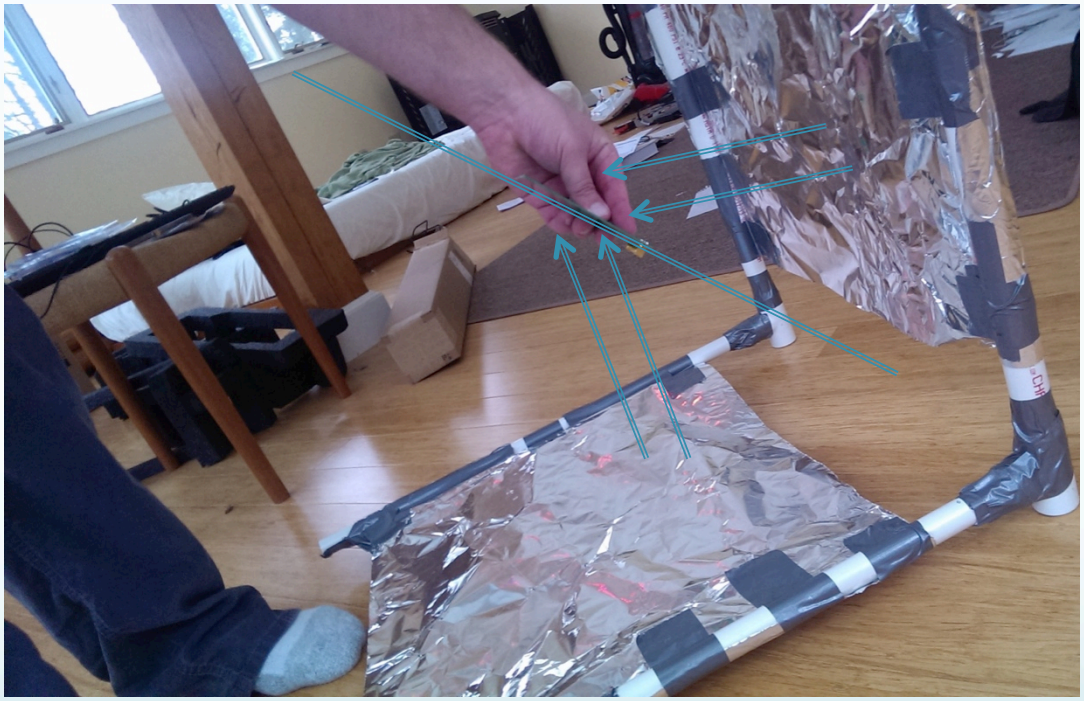
- SW:
  - GUI works fine but is a WIP … waiting to release when finish one or *three* last items.
  - Automated alerting vs manual methods (just link a few methods together)
    - **PLEASE CONTACT ME IF WANT CODE NOW!!!**
- HW:
  - Adding support for 900MHz - 2.2GHz:
    - 2 log-periodic & building corner reflectors for them:
      - Need cabling
    - Would result in (12) 2M channels in 50-900 and (6) 2M in 900-2200. Not exactly balanced, but ok.

POST TALK NOTE: Planning to do more work on both FFT windows (overlay on boundaries) & more work on detection (some methods I have not been aware of). Thanks for chats on this with: @avian2

Log Periodic (LOGI)

Of course there is work to be done on the planar nature of the reflectors

Need to mount on boom



Tough to see but the log-periodic is in my hand

Next **Bigger** Project

- World Wide Sense Network:
  - Create multiple RF arrays around the world
  - Connect them to perform cross location RF sense analysis
  - Goal:
    - Keep tabs on the occupancy distributions around the globe
- I INVITE YOU ALL TO HELP WITH THIS!
- **PLEASE** email me at **arr@watson.org** if interested!

Thank you's

- **Dimitri Stolnikov, Steve Markgraf, Hoernchen, and Kyle Keen** for releasing librtlsdr and rtl_* tool code.
  - Much of my *slave* code is based on their work.
- Chris Eng of Veracode for supporting this project
- cvoid for discussion (sadly he had to withdraw talk!)
- t12 and aempirei for fruitful early discussion
- **You** for *listening* and hopefully *acting* on this.
  - Mailing list **radiopunks-join@lists.cw-complex.com**

References

1. RTL-SDR Osmocom, http://sdr.osmocom.org/trac/wiki/rtl-sdr
2. Pyrtlsdr, https://github.com/roger-/pyrtlsdr
3. GNU Radio, http://www.gnuradio.com
4. Choosing a mounting site, http://www.hdtvprimer.com/antennas/siting.html
5. Antennas & Propagation, http://www.ccs.neu.edu/home/rraj/Courses/6710/S10/Lectures/AntennasPropagation.pdf
6. RTL2832u DVB-T COFDM, http://www.realtek.com.tw/products/productsView.aspx?Langid=1&PFid=35&Level=4&Conn=3&ProdID=257