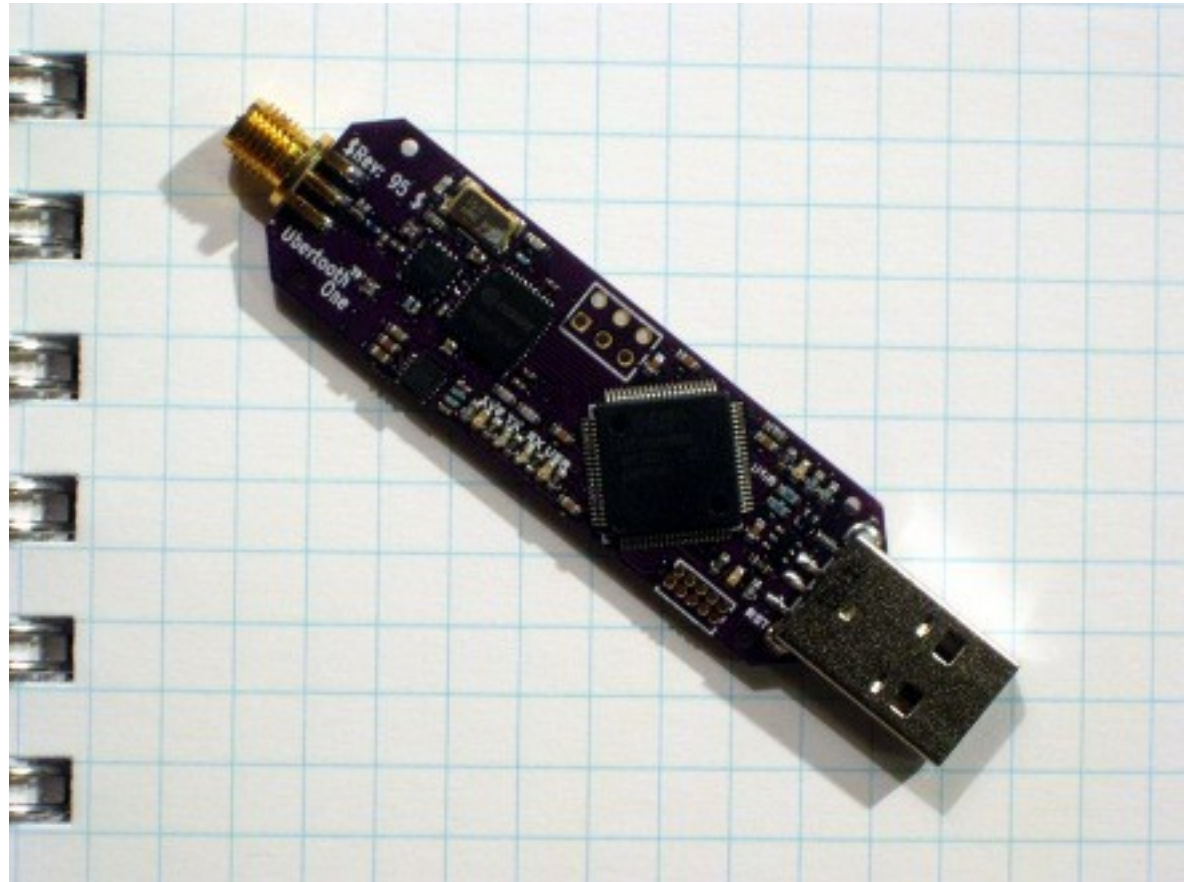


Bluetooth Packet Sniffing Using Project Ubertooth



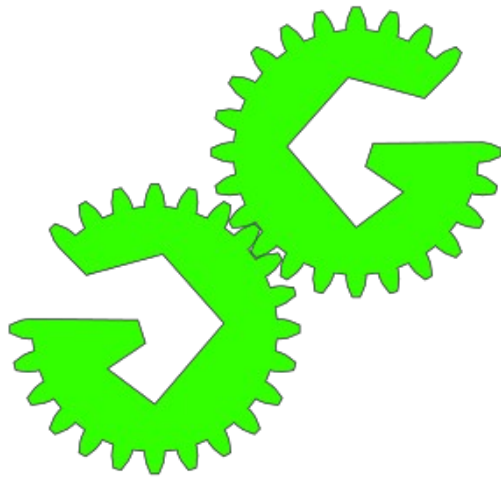
Dominic Spill
dominicgs@gmail.com

Dominic Spill



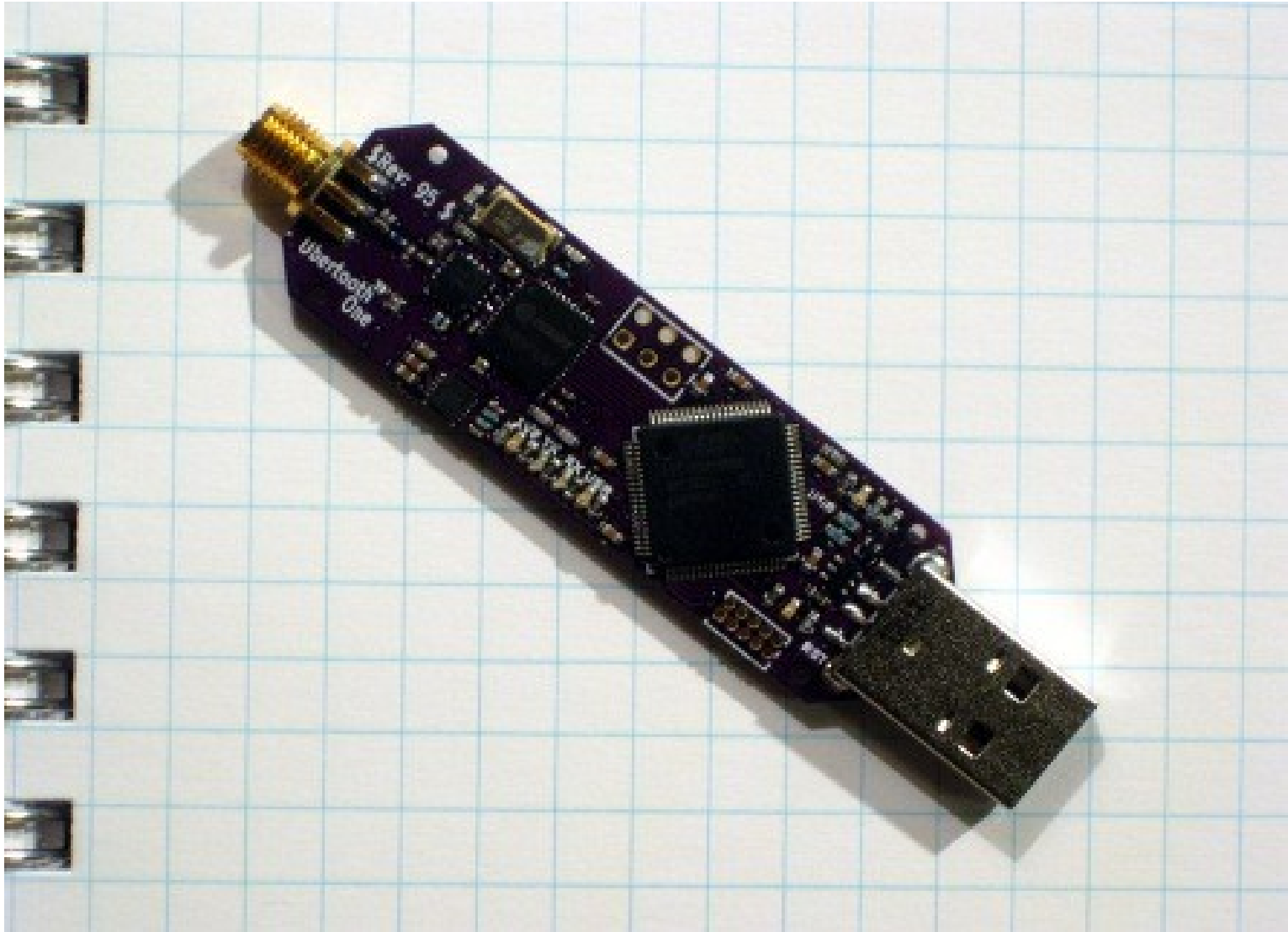
- Bluesniff: Eve meets Alice and Bluetooth
 - Usenix WOOT 07
- Building a Bluetooth monitor
 - Shmoo/Defcon/Toorcamp 09
 - With Michael Ossmann
- Lead on project Ubertoath

Disclosure



- Not an employee of GSG
- I receive some funding
- Not here to sell Ubertoos

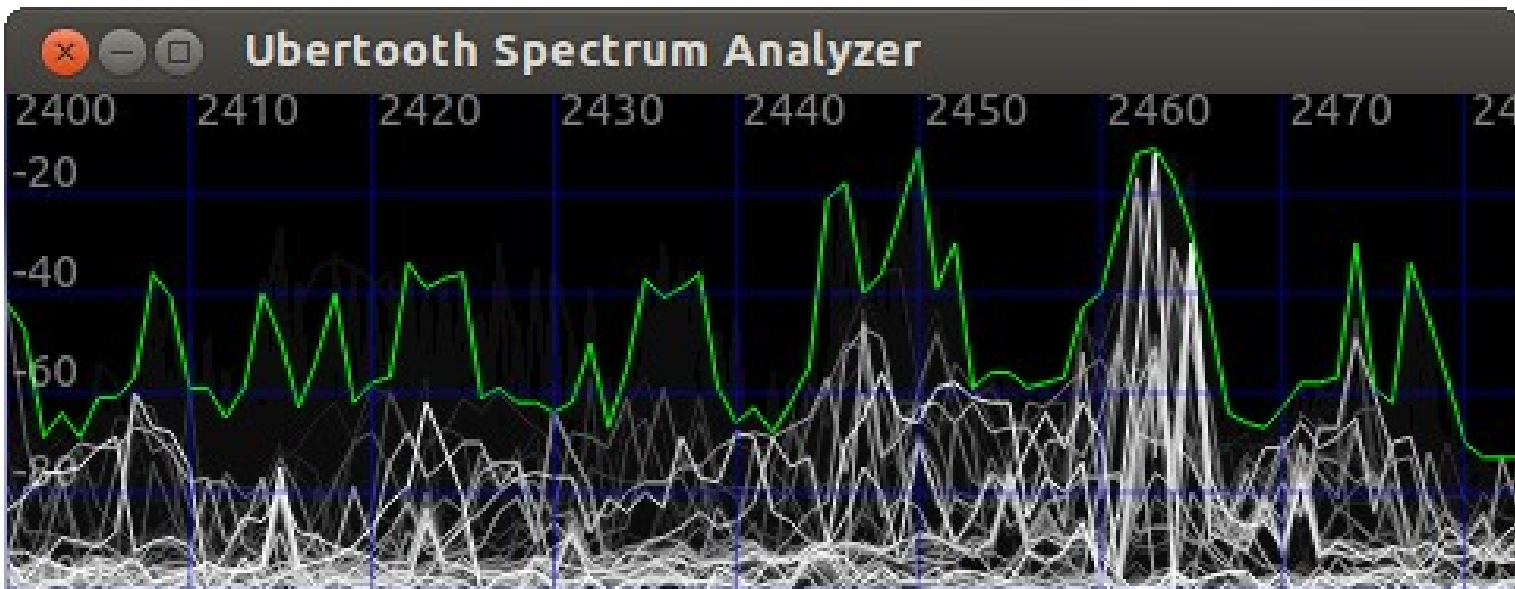
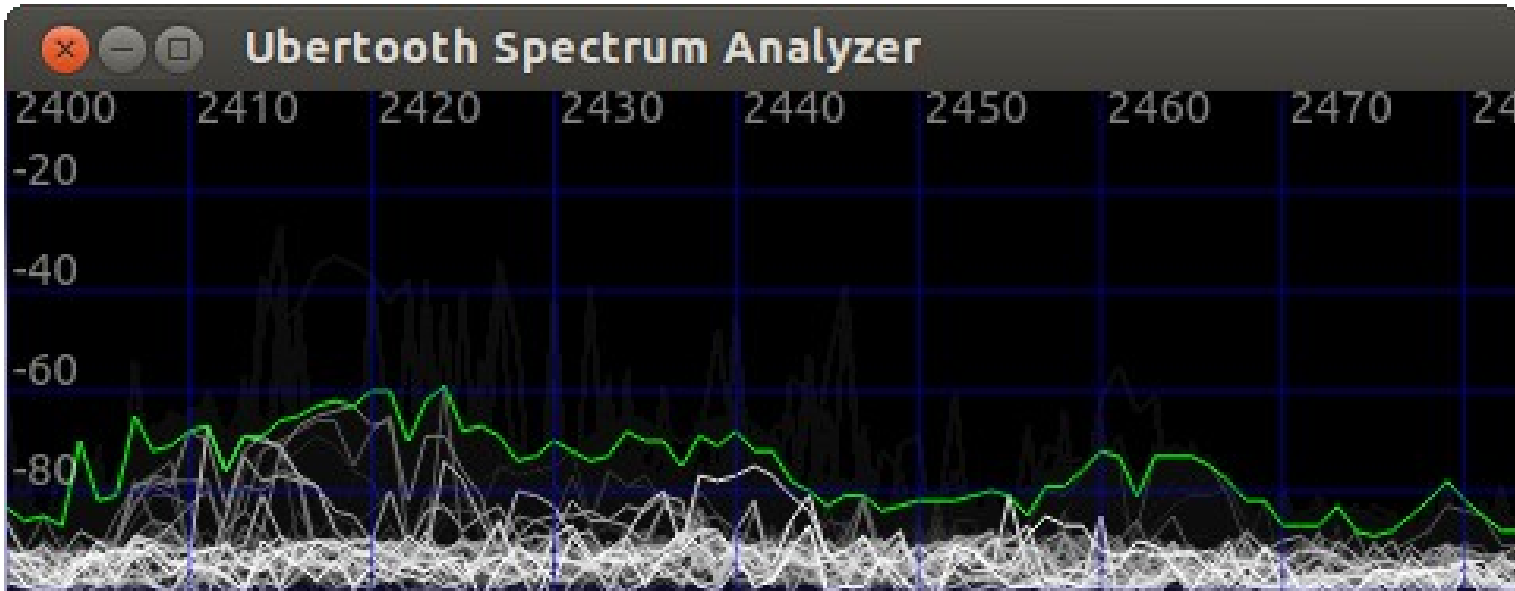
Ubertooth



Ubertooth

- Designed by Michael Ossmann
- 2.4GHz experimentation platform
- Bluetooth 1.x, Low energy, 802.11 FHSS
- Hardware
 - CC2400 (+CC2591 frontend)
 - NXP LPC1756
 - USB device (2.0)
- Open source software and hardware
 - <http://ubertooth.sourceforge.net>

Spot the difference?



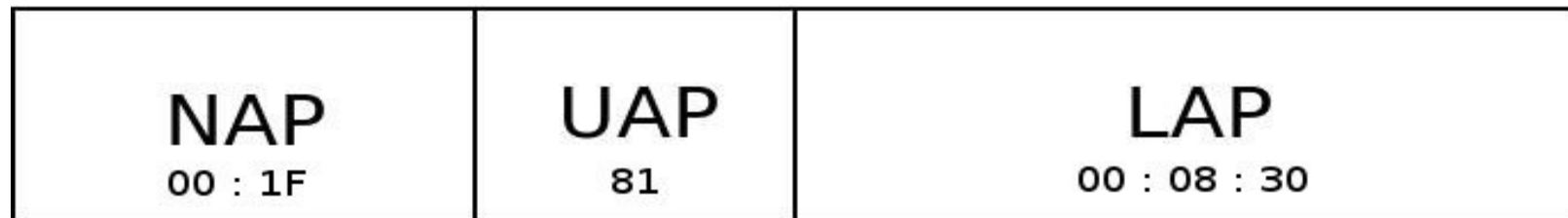
Bluetooth

Bluetooth

- 2.4GHz ISM band
- Variable data rates
 - Basic Rate – 1Mb/s
 - Enhanced Data Rate – 3Mb/s
 - High Speed - Alternate MAC/PHY – 24Mb/s
 - LE (Smart) – 200Kb/s
- FHSS @ 1600Hz
 - 79 channels

Bluetooth - Terminology

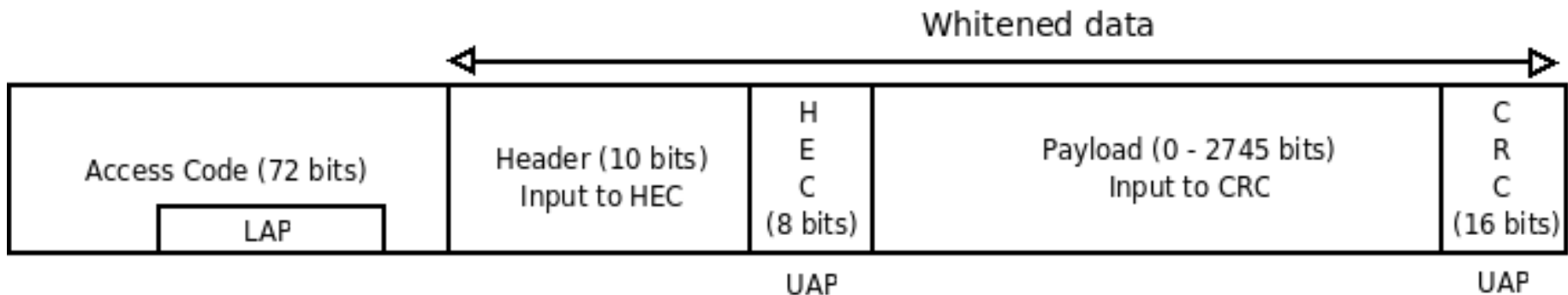
- Bluetooth device address / MAC / BD_ADDR
 - Three parts, not all present in packets
 - LAP - Lower – lowest 24 bits
 - UAP - Upper – next 8 bits
 - NAP - Non-significant – top 16 bits



- CLKN
 - 27bit 3200Hz internal clock
 - Increments twice per time slot

Bluetooth - Terminology

- Access code
 - Derived from LAP
- Packet Header
 - Error check based on UAP
- Payload
 - Possibly encrypted
 - CRC also based on UAP



Bluetooth - Terminology

- Non-Discoverable mode
 - Does not respond to inquiry scans
 - Still responds to page scans
 - Some newer devices ignore unknown page scans
- Data whitening
 - Packets XOR'd with pseudo-random sequence

Bluetooth sniffing is hard

- No “monitor mode”
 - Fixed correlator – not promiscuous
- Frequency hopping
 - 1600 hops/s
 - 625us/packet
 - Pattern based on MAC and CLKN
- Data whitening
 - PRNG initialised with CLK1-6

Finding Packets – Old method

- Find access code
 - Treat 64bit chunks as possible access codes
 - LAP stored in bits 34-57
- Check access code
 - Check trailer (2 errors)
 - Generate access code from LAP
 - Compare access code to 64bit chunk (6 errors)

Flaws

- Slow on desktop CPU
- Unworkable on low power devices
- No errors allowed in LAP
- No error correction

Error Correction

Error Correction

- (64, 30) expurgated block code
 - Based on BCH (63, 30) code
 - Calculate syndromes to find error vectors
- Supposed to correct up to 6 bit errors
 - Too many false positive results
 - Realistically correct <4 bit errors

Error Correction

- Manufacturers don't implement it
 - Known access code loaded into correlator
 - Compared to received bits
 - Up to 6 bit errors
- This is what we do for a known address

Finding Packets – New Method

- Pre-calculate syndromes for n-bit errors
 - Use known access code
 - XOR with all possible n-bit error vectors
 - Generate syndrome for each error
 - Store in hash (uthash rules!)
- For each 64bit block
 - Calculate syndrome
 - Check hash for error vector
 - Correct error

Finding Packets – New Method

Demo

Frequency Hopping

Frequency Hopping – Local Device

- Ubertooth-follow
 - Follow a local Bluetooth device
 - Use bluez to extract CLKN
 - Upload to Ubertooth
 - Start hopping

- Demo

Frequency Hopping – Local Device

- Pros
 - Reliable
 - Potentially sniff pairing
- Cons
 - Master device must be local
 - No AFH support
 - Expected soon
 - Clock drift causes problems
 - This is fixable

Frequency Hopping – Any Device

- Derive CLKN from received packets
 - Calculate hopping pattern for known address
 - Sniff single channel or hop randomly
 - Observe packets, timing and channel
 - Place packets in hopping pattern
 - Yields unique CLKN
- Calculate clock offset from CLKN → Ubertooth
- Send to Ubertooth
- Follow hopping piconet

Frequency Hopping – Any Device

- Ubetooth-hop
 - Follow a remote piconet
 - Given LAP and UAP
 - Finds clock offset and hops

- Demo

Caveats

- Adaptive Frequency Hopping
 - Coming soon
- Encryption
 - PIN – BTCrack
 - SSP – Much harder
 - Need to sniff pairing for both

Kismet Plugin

- Plugin for current and upcoming Kismet
- Dual mode
 - Survey – static or hopping
 - Follow
- Demo

Wireshark Plugin

Demo

Bluetooth Smart

- AKA
 - Bluetooth Low Energy
 - Bluetooth 4.0
 - Wibree
- Much simpler protocol
- Mike Ryan has just started working on this
 - Sniffing connection phase
 - Sniffing some data

Thanks to...

- Michael Ossmann
- Jared Boone
- Mike Kershaw (dragorn)
- “Will Code”
- Mike Ryan

Questions?

dominicgs@gmail.com

Twitter: [dominicgs](#)

Slides: dominicspill.com/ruxcon/2012.pdf