Bluetooth:

With Low Energy Comes Low Security

Mike Ryan
iSEC Partners

USENIX Security / WOOT
Aug 13, 2013

Outline

- What is Bluetooth Low Energy?
- Protocol overview
- Sniffing Techniques
- [In]security
- Injection

Outline

- What is Bluetooth Low Energy?
- Protocol overview
- Sniffing Techniques
- [In]security
- Injection

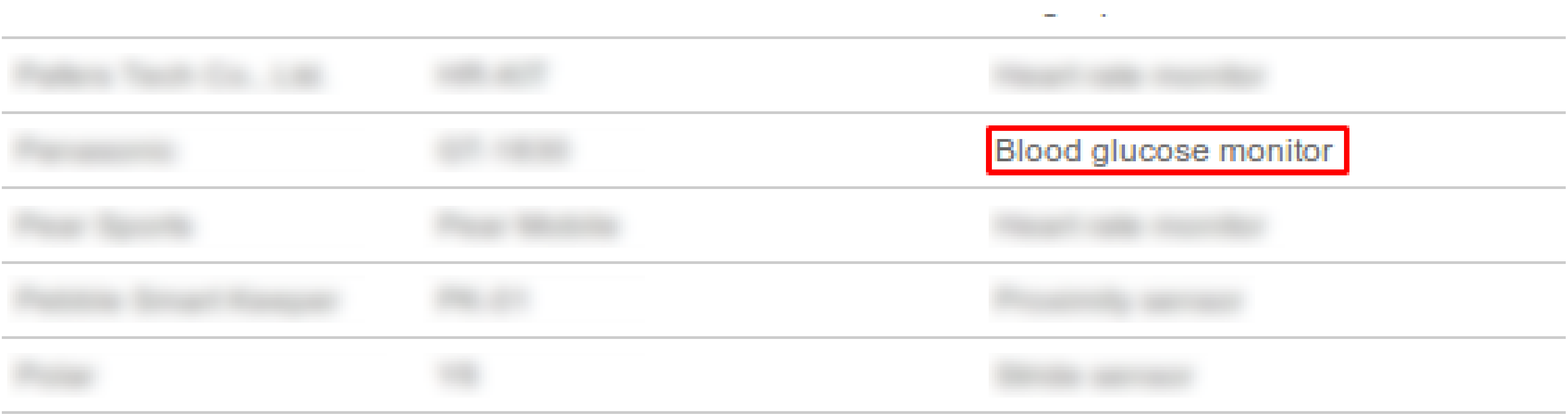
What is Bluetooth Low Energy?

What is Bluetooth ~~Low Energy~~ Smart?

- New modulation and link layer for low-power devices
- vs classic Bluetooth
  - Incompatible with classic Bluetooth devices
  - PHY and link layer almost completely different
  - High-level protocols reused (L2CAP, ATT)
- Introduced in Bluetooth 4.0 (2010)
- AKA BTLE

Where is BTLE?

- → High end smartphones
- → Sports / fitness devices
- → Door locks
- → Upcoming medical devices



By The Numbers

- 186% YoY Growth for H1 2013[1]
- "over 7 million Bluetooth Smart ICs were estimated to have shipped for use in sports and fitness devices in the first half of 2013 alone"
- "Analysts Forecast Bluetooth Smart to Lead Market Share in Wireless Medical and Fitness Devices"[2]

[1] http://www.bluetooth.com/Pages/Press-Releases-Detail.aspx?ItemID=170
[2] http://www.bluetooth.com/Pages/Press-Releases-Detail.aspx?ItemID=165

Outline

- What is Bluetooth Low Energy?
- Protocol overview
- Sniffing Techniques
- [In]security
- Injection

Protocol Stack

GATT
ATT
L2CAP
Link Layer
PHY

PHY Layer

- GFSK, +/- 250 kHz, 1 Mbit/sec
- 40 channels in 2.4 GHz
- Hopping

Physical Channels

- Advertising: 3 channels
- Data: 37 channels

RF Channel	RF Center Frequency	Channel Type	Data Channel Index	Advertising Channel Index
0	2402 MHz	Advertising channel		37
1	2404 MHz	Data channel	0	
2	2406 MHz	Data channel	1	
...	...	Data channels	...	
11	2424 MHz	Data channel	10	
12	2426 MHz	Advertising channel		38
13	2428 MHz	Data channel	11	
14	2430 MHz	Data channel	12	
...	...	Data channels	...	
38	2478 MHz	Data channel	36	
39	2480 MHz	Advertising channel		39

Hopping

- Hop along 37 data channels
- One data packet per channel
- Next channel ≡ channel + hop increment (mod 37)
- Time between hops: hop interval

3 → 10 → 17 → 24 → 31 → 1 → 8 → 15 → ...
hop increment = 7

Link Layer

LSB MSB

Preamble	Access Address	PDU	CRC
(1 octet)	(4 octets)	(2 to 39 octets)	(3 octets)

Figure 2.1: Link Layer packet format

- PDU min of 2 bytes due to 2 byte header
- LLID: Control vs Data
- Length

L2CAP and Beyond

- Use existing decoders for this
- Not a Hard Problem™

Recap

GATT
ATT
L2CAP
Link Layer
PHY

Outline

- →What is Bluetooth Low Energy?
- →Protocol overview
- →Sniffing Techniques
- →[In]security
- →Injection

sniffing
Bluetooth
is
hard

sniffing

Bluetooth LE

is slightly

less hard

How do we sniff it?

Start at the bottom and work our way up:

PC → GATT

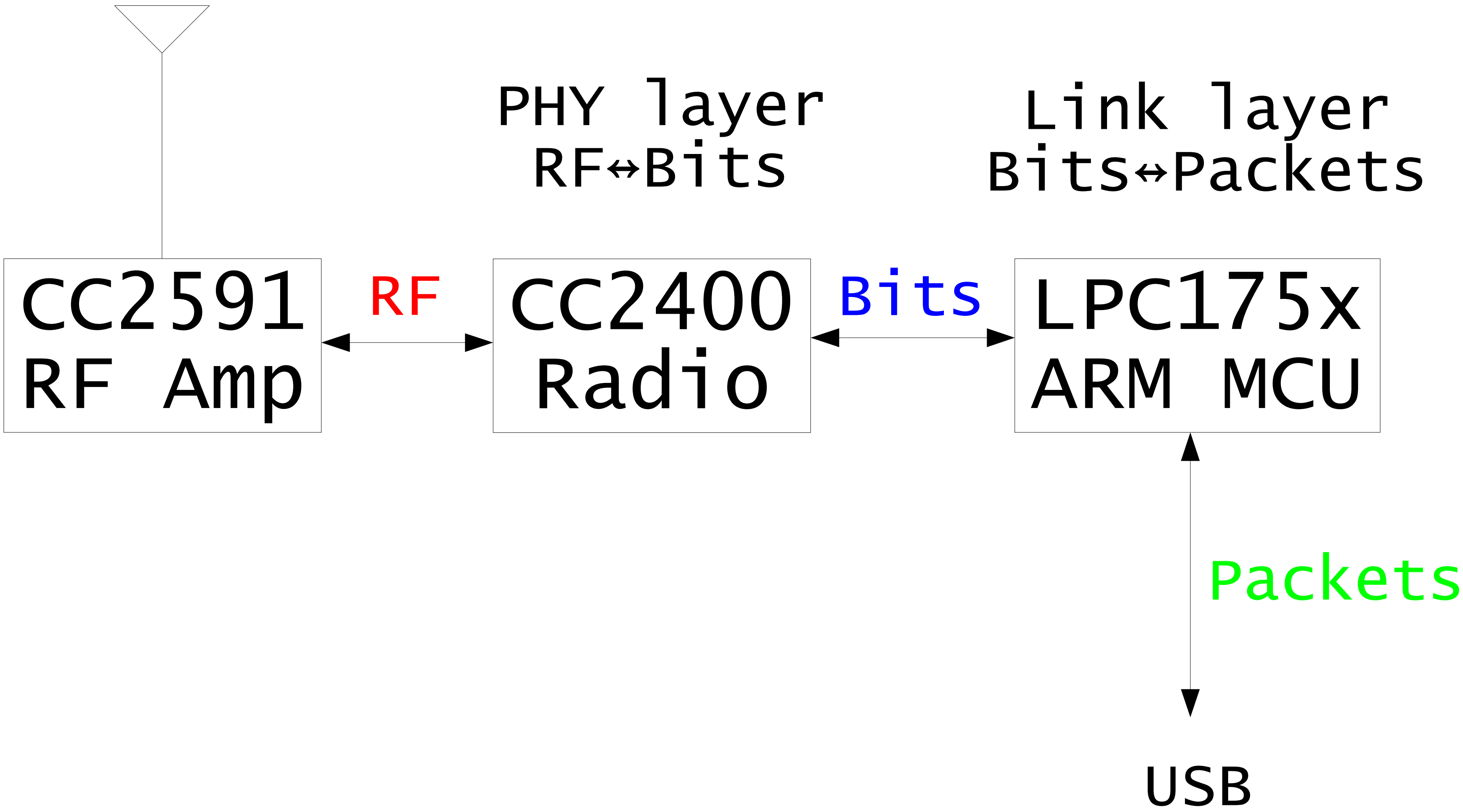
PC → ATT

PC → L2CAP

Ubertooth → Link Layer

Ubertooth → PHY

Ubertooth Block Diagram



Capturing: PHY Layer

- Configure CC2400
  - Set modulation parameters to match Bluetooth Smart
  - Tune to proper channel
- Follow connections according to hop pattern
  - Hop increment and hop interval, sniffed from connect packet or recovered in promiscuous mode
- Hand off bits to ARM MCU

What Info Do We Need?

Preamble (1 octet)	Access Address (4 octets)	PDU (2 to 39 octets)	CRC (3 octets)

- Access Address
  - Advertising: Fixed `0x8E89BED6`
  - Connection: Varies
- Channel number
  - Hop interval
  - Hop increment
- Nice to have: CRCInit

Where?

Easy mode: Connect packet!

Link Layer

LSB			MSB
Preamble (1 octet)	Access Address (4 octets)	PDU (2 to 39 octets)	CRC (3 octets)

Figure 2.1: Link Layer packet format

What we know: Access Address

What we have: Sea of bits

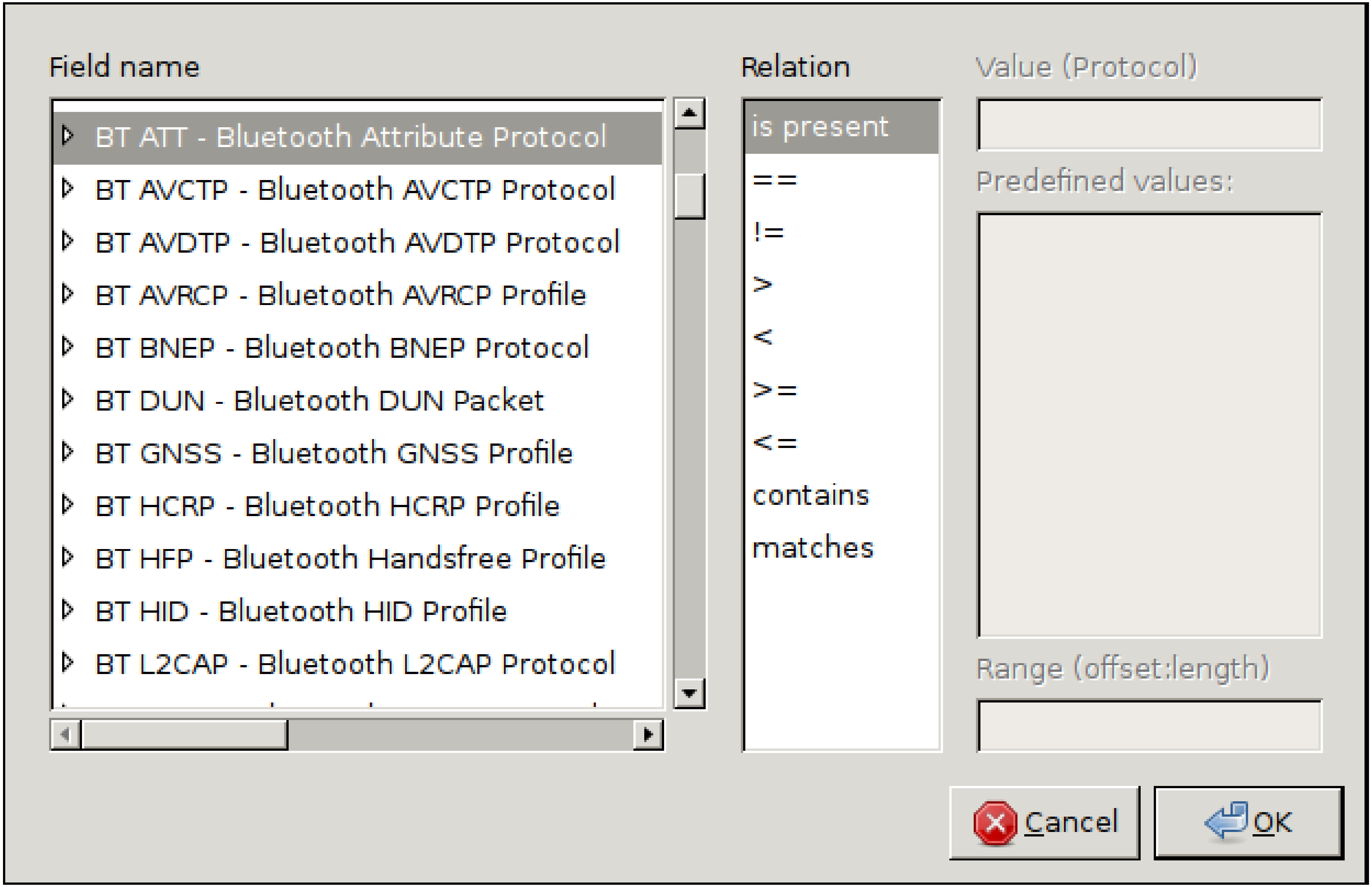
What we want: Start of PDU

CC2400 does this

FO FREE

10001110111101010101
10011100000100011001
11100100110100011101

PHY Layer.. Link Layer..

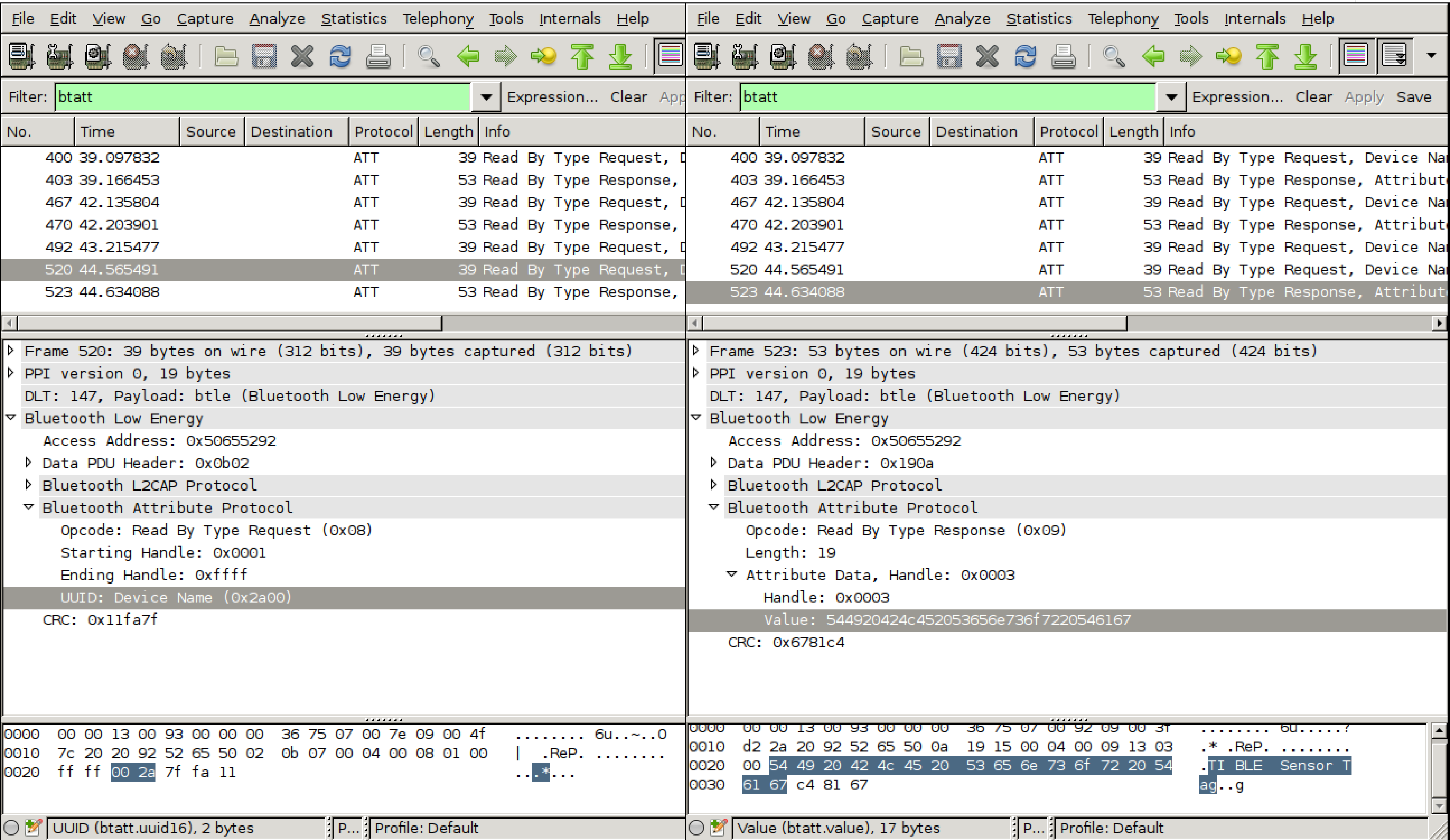


Capturing Packets... To PCAP!

- ubertooth-btle speaks packets
- libpcap → dump raw packet data
- PPI header (similar airodump-ng and kismet)

- We have a DLT for Bluetooth Smart
  - Unique identifier for the protocol
  - Public release of Wireshark plugin Coming Soon™

Wireshark Awesomeness



Promiscuous Mode

- Techniques for recovering
  - Access Address
  - CRCInit
  - Hop Interval
  - Hop Increment

Recovering Access Address

- Sit on data channel waiting for empty data packets
- Collect candidate AA's and pick one when it's been observed enough

10001110111101010101
10011100000100011001
10000000000000001101
10100011000110000101

Not depicted: whitening!

Recovering CRCInit

- Filter packets by Access Address
- Plug CRC into LFSR and run it backward

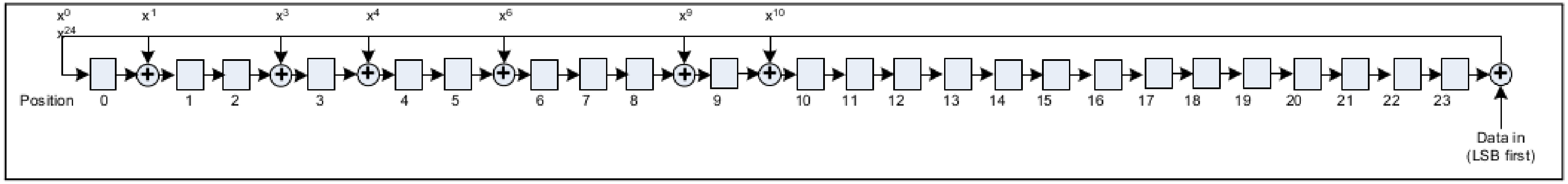


Figure 3.2: The LFSR circuit generating the CRC

See also "Bluesniff: Eve meets Alice and Bluetooth", USENIX WOOT '07

Recovering Hop Interval

- Observation: 37 is prime
- Sit on data channel and wait for two consecutive packets

$$\frac{\Delta t}{37} = hop\ interval$$

Recovering Hop Increment

- Start on data channel 0, jump to data channel 1 when a packet arrives
- We know hop interval, so we can calculate how many channels were hopped between 0 and 1

Recovering Hop Increment (math)

$$0 + \text{hopIncrement} \times \text{channelsHopped} \equiv 1 \pmod{37}$$

$$\text{hopIncrement} \equiv \text{channelsHopped}^{-1} \pmod{37}$$

$$\text{channelsHoppped}^{-1} \equiv \text{channelsHopped}^{37-2} \pmod{37}$$

We use a LUT to convert that to hop increment

Sniffing Summary

- Connection following
- Promiscuous: Recovering the four values
  - Access address
  - CRCInit
  - Hop interval
  - Hop Increment

Outline

- What is Bluetooth Low Energy?
- Protocol overview
- Sniffing Techniques
- [In]security
- Injection

Encryption

- → Provided by link layer
- → Encrypts and MACs PDU
- → AES-CCM

LSB MSB

Preamble (1 octet)	Access Address (4 octets)	PDU (2 to 39 octets)	CRC (3 octets)

Figure 2.1: Link Layer packet format

Custom Key Exchange Protocol

- Three stage process
- 3 pairing methods
  - Just Works™
  - 6-digit PIN
  - OOB

- “None of the pairing methods provide protection against a passive eavesdropper” -Bluetooth Core Spec

Cracking the TK

```
confirm
=
AES(TK, AES(TK, rand XOR p1) XOR p2)

GREEN = we have it
  RED = we want it
```

TK: integer between 0 and 999,999

Just Works™: always 0!

Cracking the TK – With *crackle*

Total time to crack:
< 1 second

And That's It

- TK → STK
- STK → LTK
- LTK → Session keys

KEY EXCHANGE = BROKEN

100% PASSIVE

LTK Reuse

- Good for security: pair in a faraday cage
- Counter-mitigation: Active attack to force re-pairing

Decrypting

- Assumption: Attacker has LTK – reused!
- Procedure
  - Attacker passively capturing packets
  - Connection established
  - Session information captured

Decrypting – With *crackle*

- Yes, crackle does that too!
- crackle will decrypt
  - a PCAP file with a pairing setup
  - a PCAP file with an encrypted session, given an LTK

Am I Affected?

- Probably
- Exception: Some vendors implement their own security on top of GATT
  - Did they talk to a cryptographer?

Security Recap

- Key exchange broken
- LTK reuse means all communication is effectively compromised
- 99% passive
  - Worst case scenario: one active attack with off-the-shelf hardware

Outline

- What is Bluetooth Low Energy?
- Protocol overview
- Sniffing Techniques
- [In]security
- Injection

Injection

- Pretty much the same as receiving, opposite direction
- Follow the spec!
  - Link layer header
  - Payload data
- Hand that off to Ubertooth
  - Calculate CRC
  - Whiten
- Devil is in the CC2400 details

Demo

- D
  - e
    - m
      - o

Capabilities

- Ubertooth
  - Passively intercept Bluetooth Smart
  - Promiscuous mode
  - Injection
- Wireshark plugins
- crackle
  - Crack TK's sniffed with Ubertooth
  - Decrypt PCAP files with LTK

Software

- Ubertooth and libbtbb
  - http://ubertooth.sourceforge.net/
- crackle
  - http://lacklustre.net/projects/crackle/
- nano-ecc (8-bit ECDH and ECDSA)
  - https://github.com/iSECPartners/nano-ecc

Thanks

Mike Ossmann
Dominic Spill

Mike Kershaw (dragorn)
#ubertooth on freenode
bluez
Bluetooth SIG

USENIX
iSEC Partners

Thank You

Mike Ryan

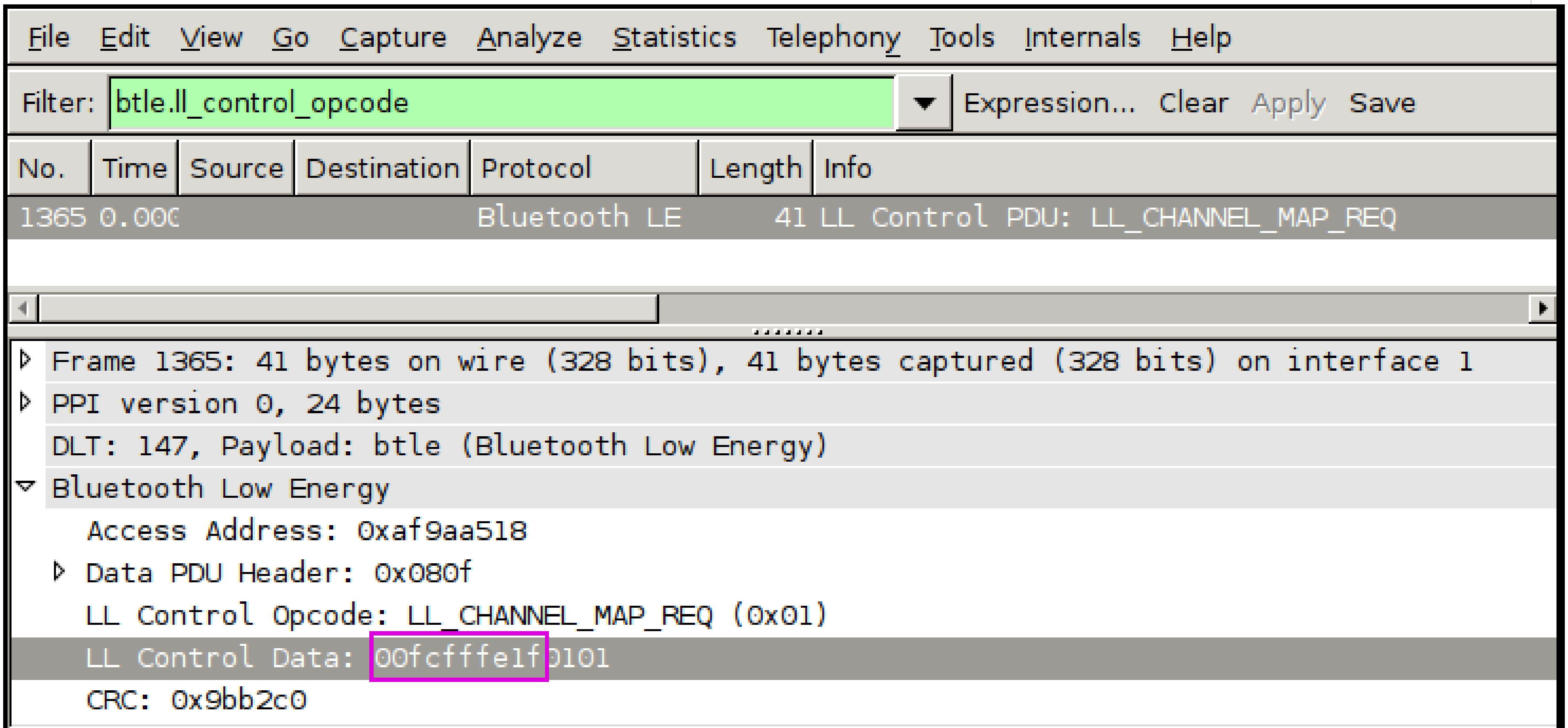
iSEC Partners

@mpeg4codec

mikeryan@isecpartners.com

http://lacklustre.net/

Apocrypha (extra)



0x1FFEFFFC00: remove channels 12, 27-36

Encryption Mitigation (extra)

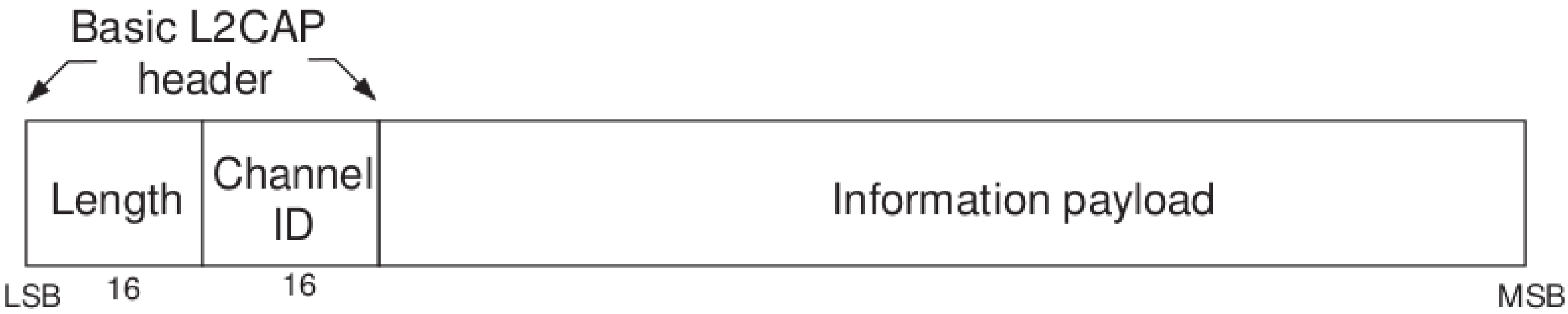
- Every session uses a different session key
- Every session uses several nonces

  Solution: jam the connection to restart a session

- LTK exchanged once, used many times

  Solution: inject LTK_REJECT_IND message

L2CAP (extra)



Basic L2CAP header
Length
Channel ID
Information payload
LSB
16
16
MSB

ATT/GATT (extra)

- Services: groups of characteristics
- Characteristics
  - Operations
- Everything identified by UUID
  - 128 bit
  - Sometimes shortened to 16 bits

Example GATT Service: Heart Rate (extra)

- Service: `0x180D`
- Characteristic 1: `0x2A37` – Heart Rate
  - Can't read or write
  - Notify: subscribe to updates
- Characteristic 2: `0x2A38` – Sensor Location
  - Readable: 8 bit int, standardized list
- Other characteristics: `0x2803`, `0x2902`, ...