A Robust Algorithm for Sniffing BLE Long-Lived Connections in Real-time

Sopan Sarkar, Jianqing Liu & Emil Jovanov

Department of Electrical and Computer Engineering
University of Alabama in Huntsville

December 12, 2019

Outline

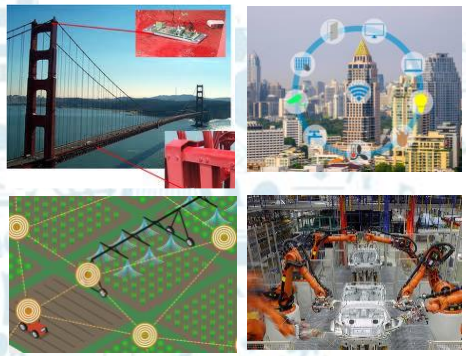
Outline

Background

Internet of Things (IoT)

Smart Health

Smart Transportation

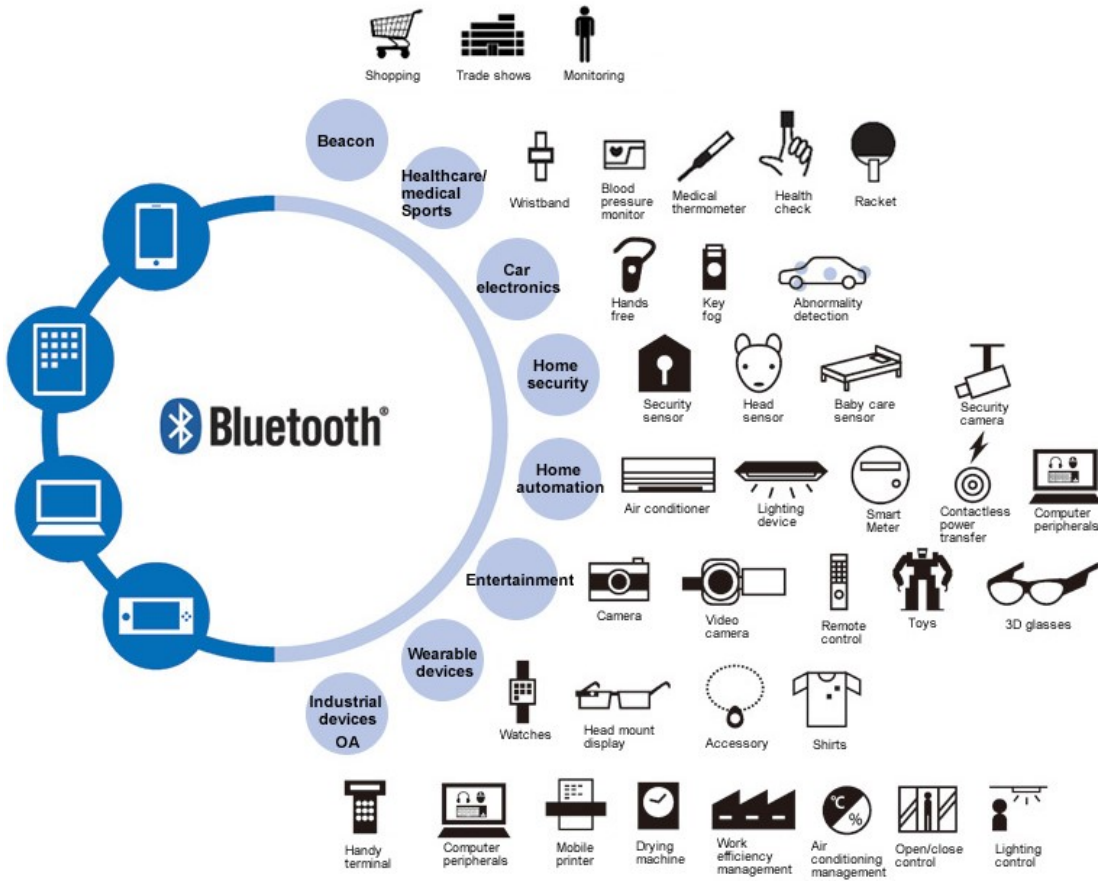


Smart Industry

Smart Energy

Background

Bluetooth Low Energy (BLE) in IoT



Background

BLE Preliminaries

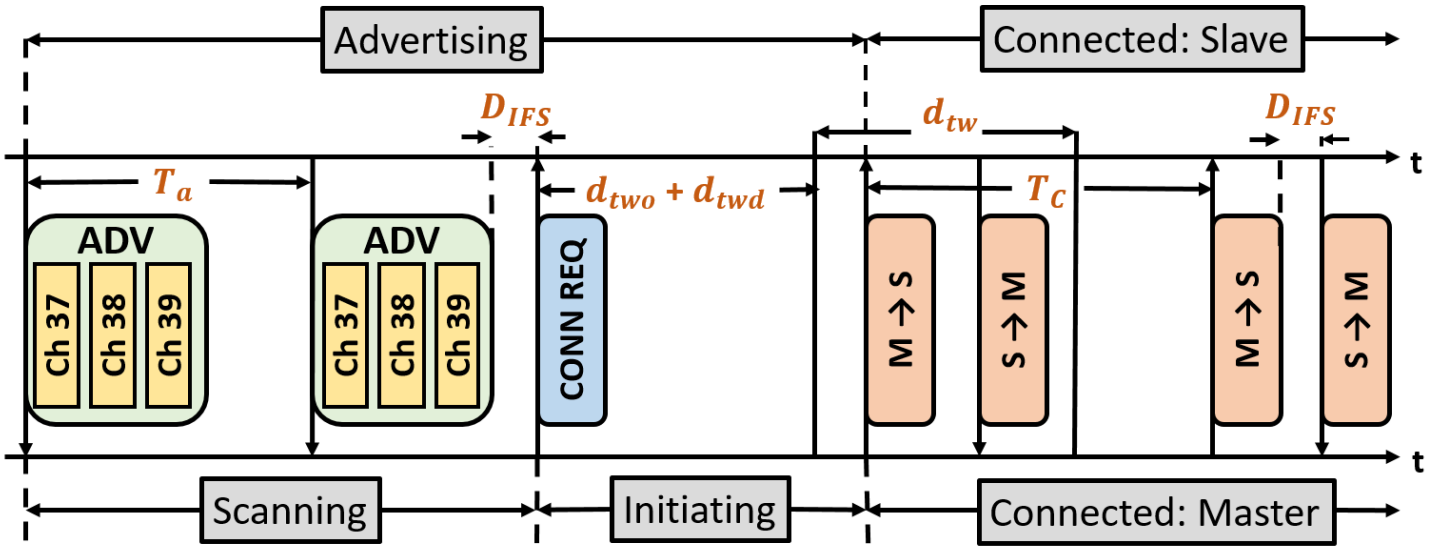
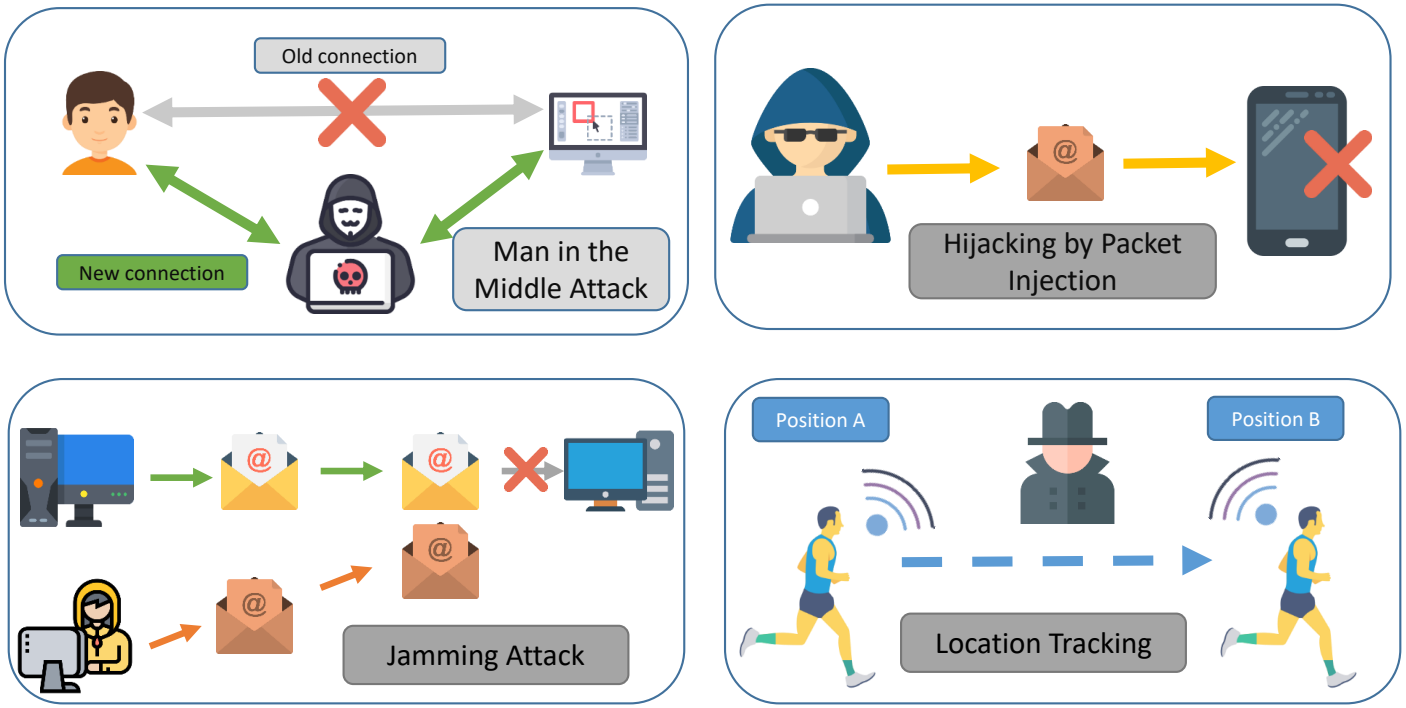


Figure: BLE Link Layer states

- Connection parameters(Access address, Hop interval, Channel map and Hop increment) are exchanged during "initiating state"
- Adaptive frequency hopping (AFH) - Channel map updates based on the channel condition

BLE Security & Privacy

BLE Security - Current State



- BLE suffers from various attacks and can be detected by intrusion detection system;
- Not well investigated paradigm - BLE sniffing

Challenges

Sniffing BLE is hard!

- Wide-band receivers are bulky and costly!
- Three parameters needed to follow a BLE connection - hop interval, hop increment and channel map are exchanged during Advertisement phase
- In connected state BLE use Frequency hopping spread spectrum and AFH

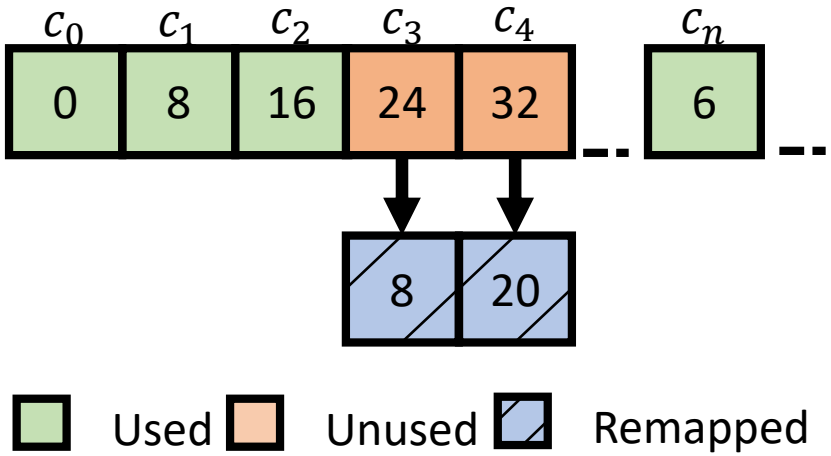


Figure: BLE channel selection algorithm

Contributions

Our contributions -

- Developed algorithm to sniff BLE in connected state
- We used a single-channel and low-cost radio sniffer Ubertooth One for implementation and evaluation
- Our algorithm runs in real-time and is prone to AFH of BLE
- Proposed algorithm can calculate AFH parameters with 96% accuracy and sniff more than 80% transmitted data

Outline

Adversary Model

System Model

- We consider an indoor environment
- Most of the BLE devices are in connected state
- Some are in advertisement state

Attack Model

- Consider adversary is in close proximity to victim BLE devices
- Has limited resources in terms of equipment and computation
- Uses single-radio sniffer

Determining Hop Interval

- BLE AFH has a period of 37
- We stay on a single BLE channel and determine the time difference $\Delta t'$ between adjacent packet receptions
- We compare and analyze the set of $\Delta t'$ and find the repetitive pattern
- The sum of $\Delta t'$ in a single pattern is T which accounts for 37 hops.
- We get the connection interval as,

$$c_{int} = \frac{1}{37} \sum \Delta t \tag{1}$$

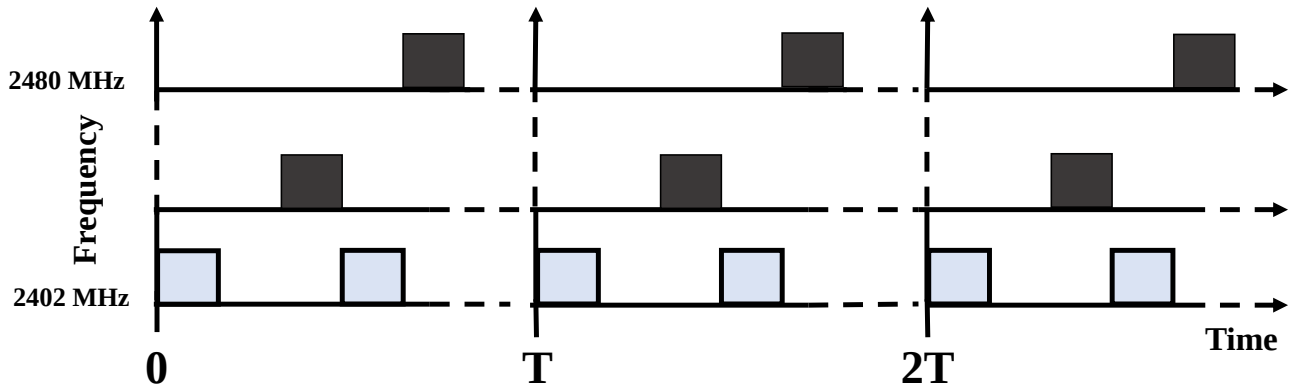


Figure: BLE frequency hopping diagram where the blue block represents the observed channel

Determining Hop Increment

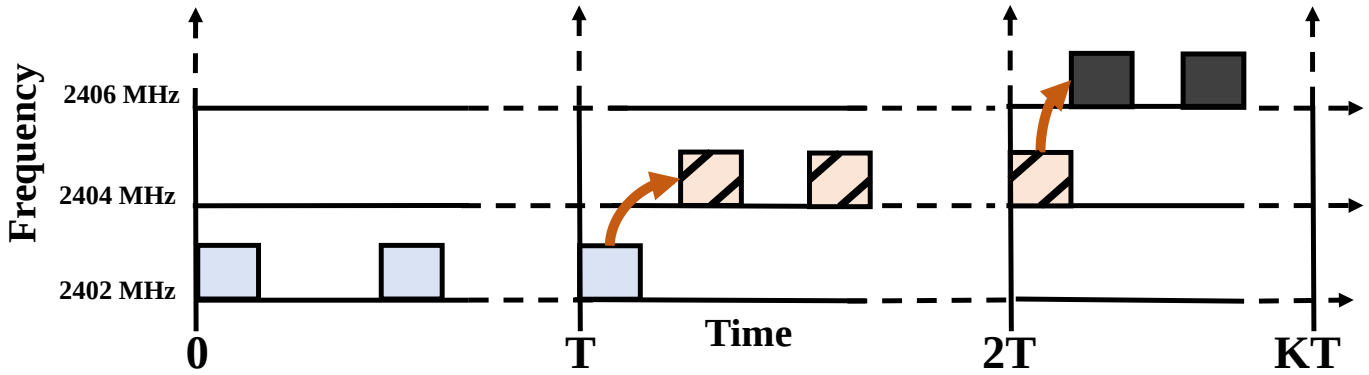


Figure: Hop increment for multiple appearances of a channel within a period

- At t_1, t_2 we observe channel c_1 and c_2, then $h = \frac{t_2 - t_1}{c_{int}}$ and

$$h_{inc} = \frac{(c_2 - c_1)^{-1}}{h} \bmod 37 \tag{2}$$

- channel map is less than 37 and channels repeat in a period
- n, m appearances of c_1, c_2 has $n \times m$ hop intervals
- Increasing channels in calculation increases probability of determining hop interval correctly - for 3 channels, probability is more than 50%

Determining Channel Map

- We hop using the channel selection algorithm,

$$c_{n-1} = (c_n + h_{inc}) \bmod 37, \tag{3}$$

- Channels in which BLE connection appeared are in the channel map
- If synchronization lost - repeat channel map detection algorithm

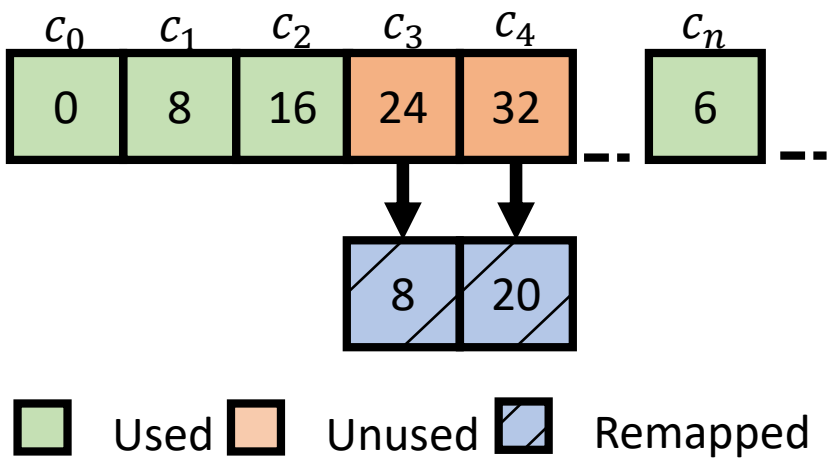


Figure: Determining channel map for BLE

Outline

Experimental Setup

Experimental setup

- We use two Ubertooth One with a Linux PC to implement the algorithm
- One Ubertooth runs in default follow mode and the other runs our algorithm
- We use Cypress BLE as test devices
- Channel map is set to update every 15s, 30s and 60s

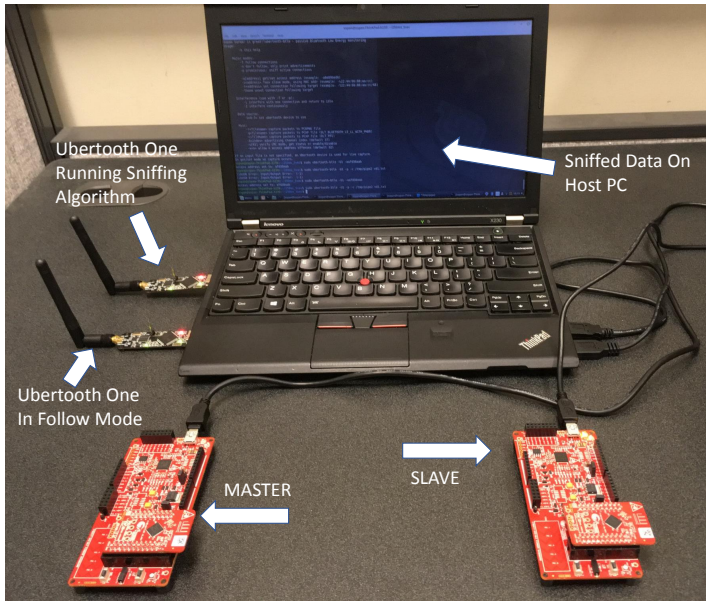


Figure: Experiment setup in an indoor controlled environment

Performance Evaluation

Latency and accuracy in determining AFH parameters

- Latency is measured in hops

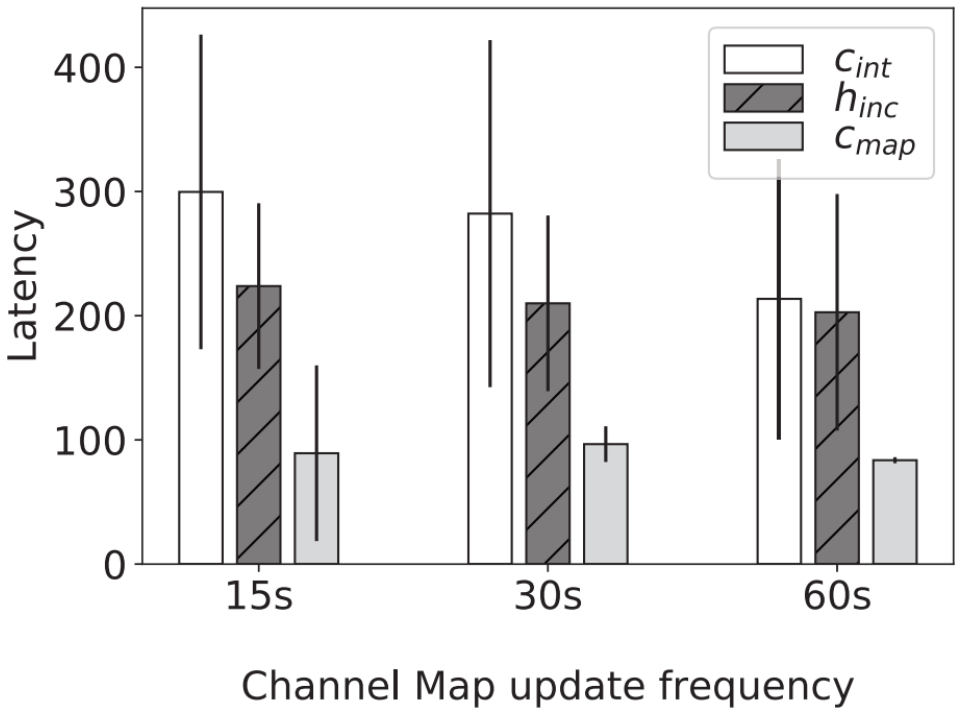


Figure: Latency in deriving BLE AFH

Parameter	15s	30s	60s
c_{int}	100%	100%	100%
h_{inc}	100%	96%	96%
c_{map}	100%	96%	96%

Table: Accuracy for determining BLE AFH parameters

Performance Evaluation

Accuracy in sniffing BLE packets

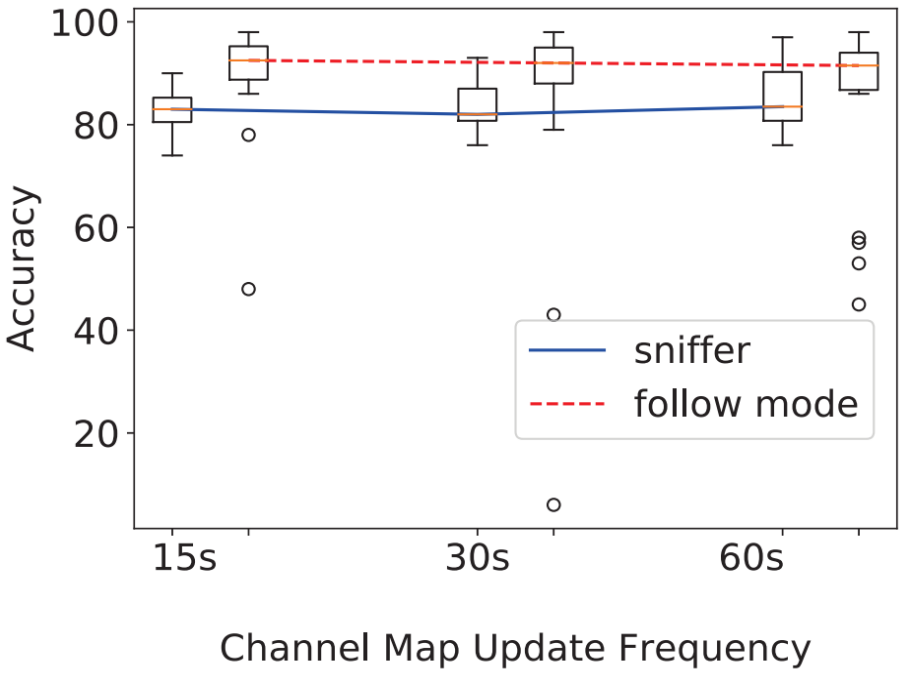


Figure: Sniffing accuracy in 100 ms data sending rate

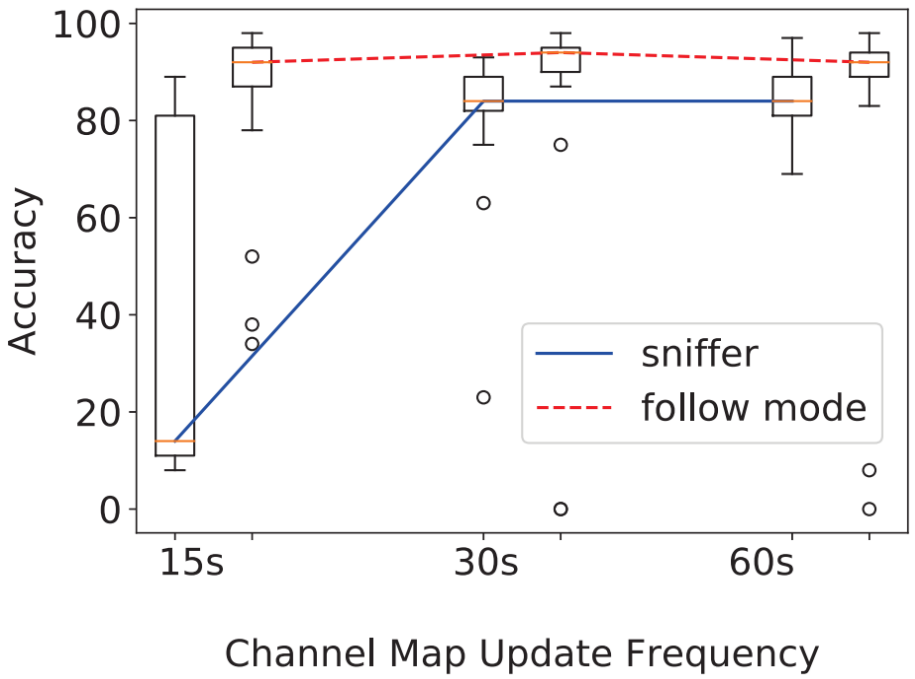


Figure: Sniffing accuracy in 250 ms data sending rate

Outline

Conclusion and Future Work

Conclusion

- We are able to sniff BLE in connected state and extract its AFH parameters
- Our algorithm didn't rely on reading the PDU data but only the access address
- We were able to calculate AFH parameters with 96% accuracy and sniff more than 80% of the transmitted data

Available at: `https://github.com/sopan-sarkar/ubertooth`

Future work

- Estimate the channel occupancy of the target BLE device and get an idea of the channel map
- Determine the access address of the target BLE device from the pool of sniffed access addresses

Thank You!