

Lecture 04

BLE Advertisement Deep Dive

CS433 – Wireless Protocols for IoT
Branden Ghena – Spring 2025

Materials in collaboration with
Pat Pannuto (UCSD) and Brad Campbell (UVA)

Announcements

- Wireshark Lab
 - Sorry that was not a great lab experience
 - We're going to try again this Friday, but in groups
 - I'll re-evaluate the plan depending on how that goes
 - Get checkoffs this week during office hours
 - Today 5-6 works
 - Last office hours before deadline is 5-7 on Thursday
- BLE packets homework will be up later tonight

Today's Goals

- Quick overview of Physical and Link layers for BLE
- Describe BLE advertising and scanning roles
- Deep dive into advertisements. Questions we might ask as researchers.
 - What are the real-world use cases of advertisements?
 - How much energy do advertisements take?
 - What is the probability of receiving a packet?
 - What is the probability of receiving data?

Basics of Bluetooth Low Energy (BLE)

- Direct device-to-device communication
 - Usually: Computer to Thing
 - Smartphone to device, Laptop to device, etc.
- Focus on making the “Thing” really low energy
 - Push energy-intensive requirements onto “Computer”
- Devices (Computer or Thing) are servers with accessible fields
 - Not the traditional send-explicit-packets interface you might be expecting
 - Lower layers are still exchanging packets to make it work



Bluetooth Classic and BLE co-exist & have different use cases

Bluetooth Classic (around since 1999)

- Continuous data communication
- Can afford higher battery power requirements



Headphones



Car BT Audio



Mobile Photo Printers



Fitness Trackers



Tracking Devices
(eg: Apple AirTag)

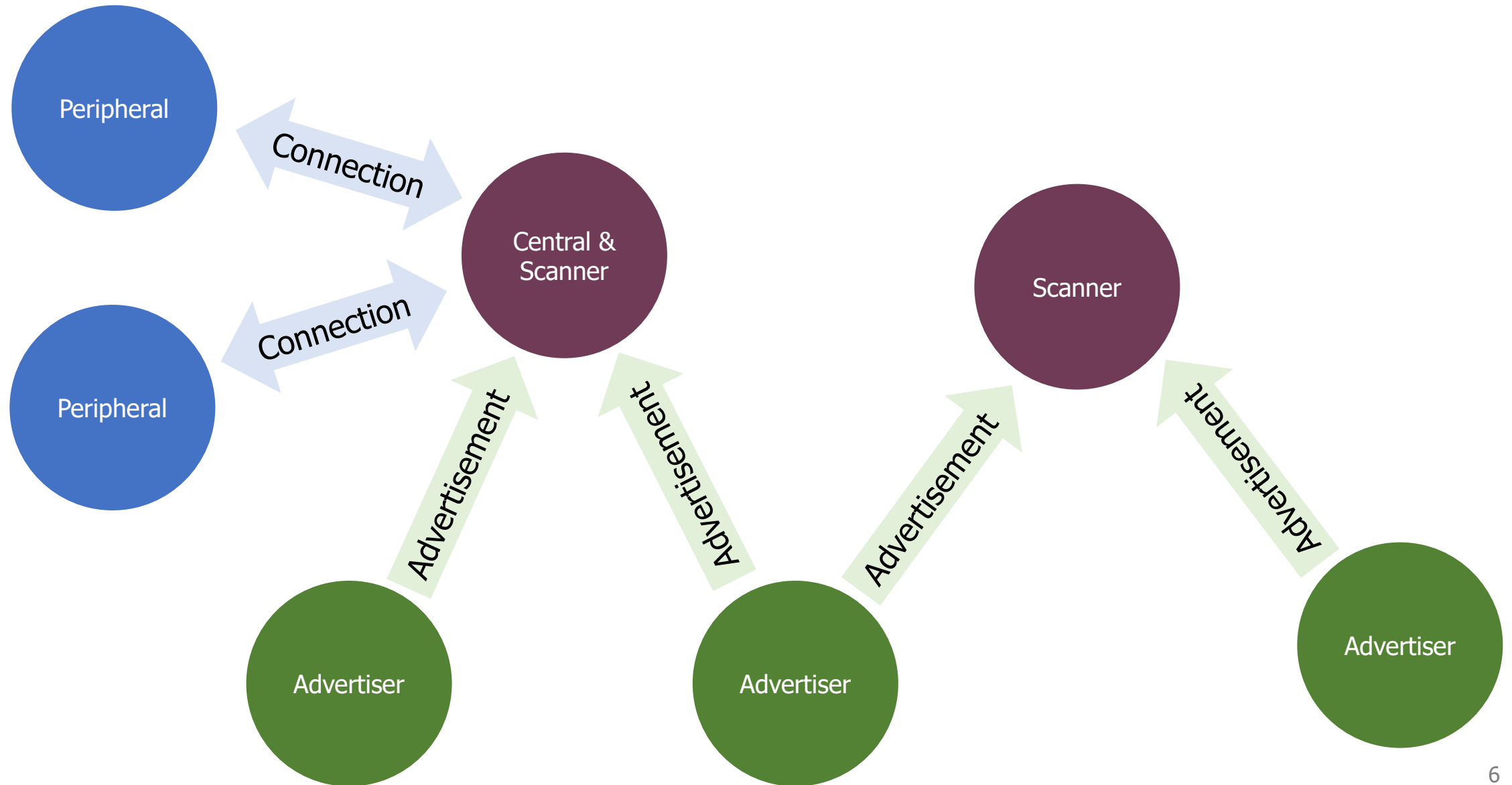


Switches
and Sensors

Bluetooth Low Energy (introduced in 2010)

- Small amounts of data transfer
- Low power requirements
 - "months or years" on small batteries
- Ideal for IoT devices

BLE network topology

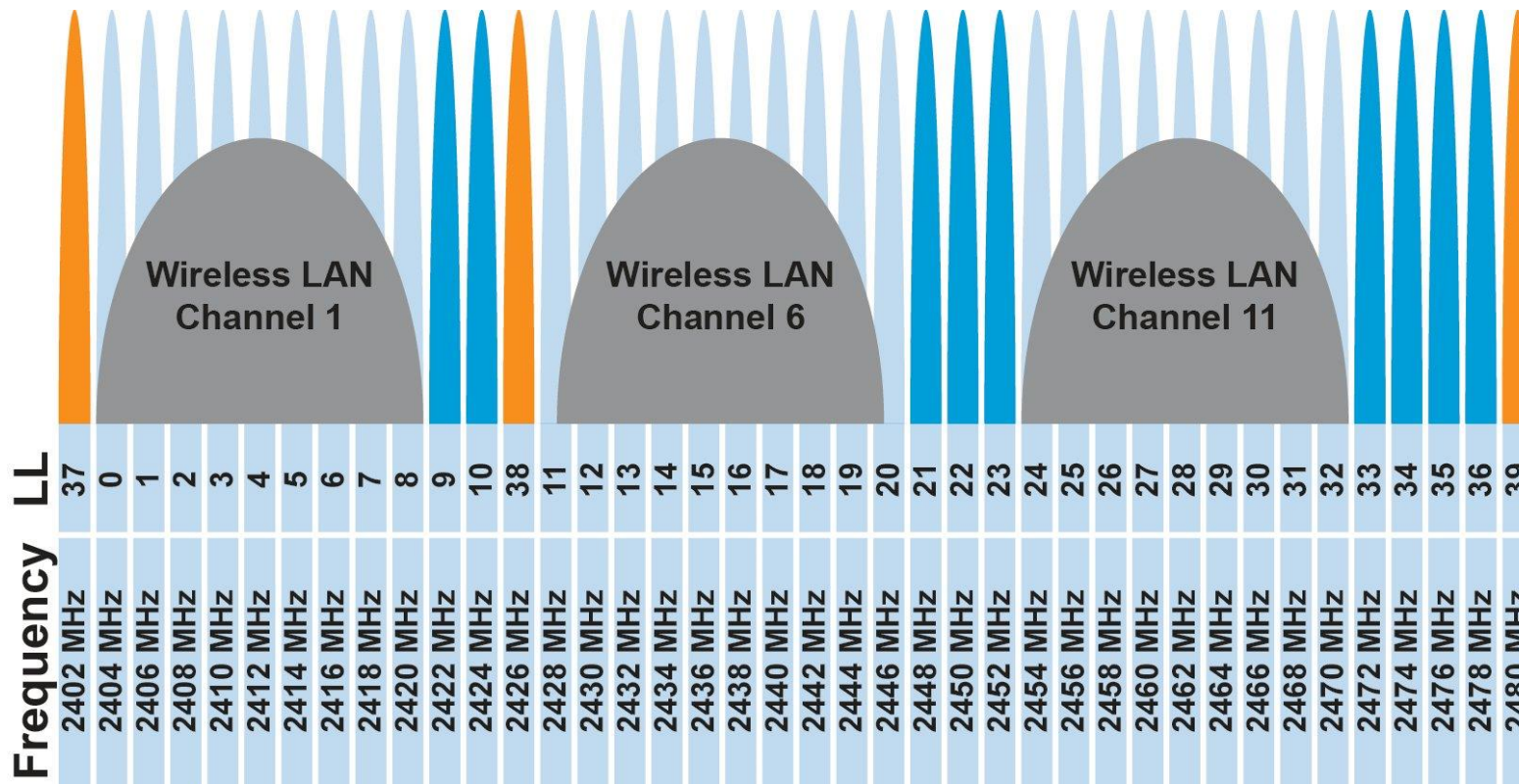


Outline

- **BLE Layers**
 - **Physical Layer**
 - Link Layer
- BLE roles
 - Advertising
 - Scanning
 - Scan Responses
- Communicating with advertisements
 - Advertisement Use Cases
 - Energy Use
 - Packet Collisions

BLE frequency

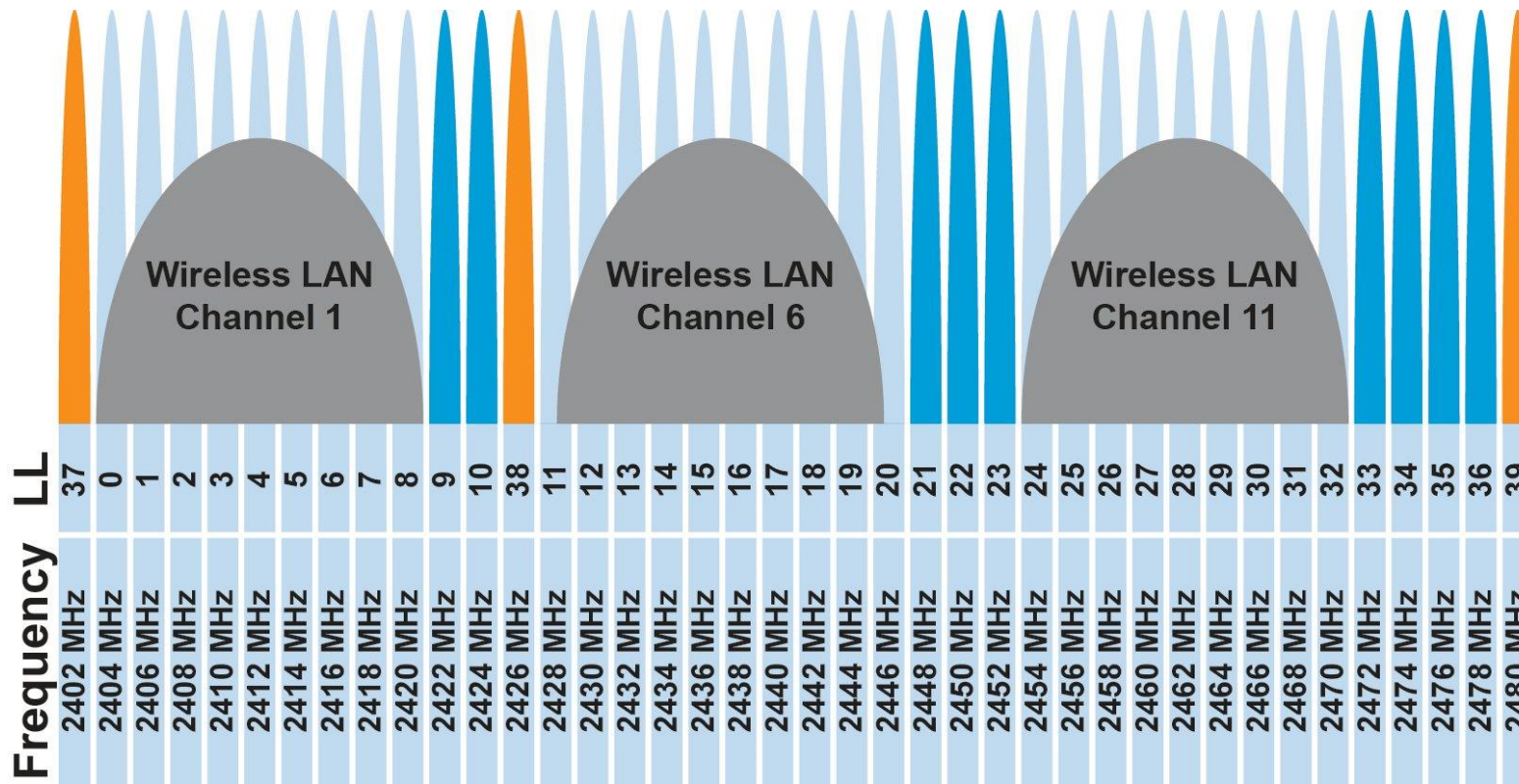
- 2.4 GHz carrier, Forty 2-MHz channels, 1 Mbps data rate
 - 37, 38, 39 for advertising
 - 0-36 for connection (FHSS)



Why doesn't BLE avoid WiFi altogether?

BLE frequency

- 2.4 GHz carrier, Forty 2-MHz channels, 1 Mbps data rate
 - 37, 38, 39 for advertising
 - 0-36 for connection (FHSS)



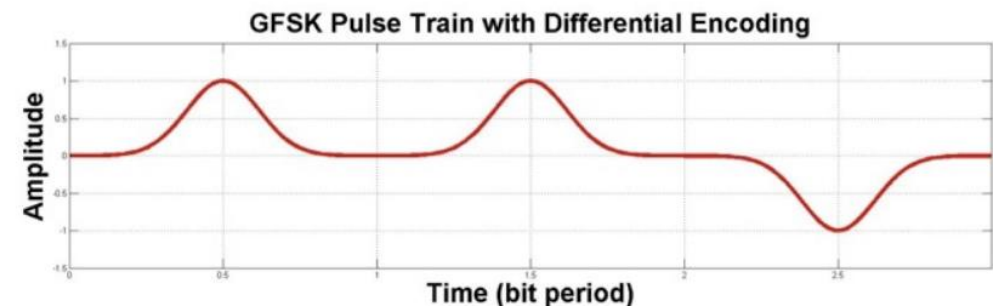
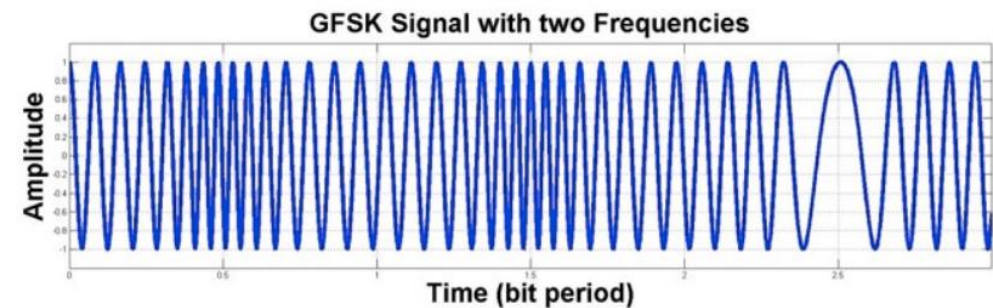
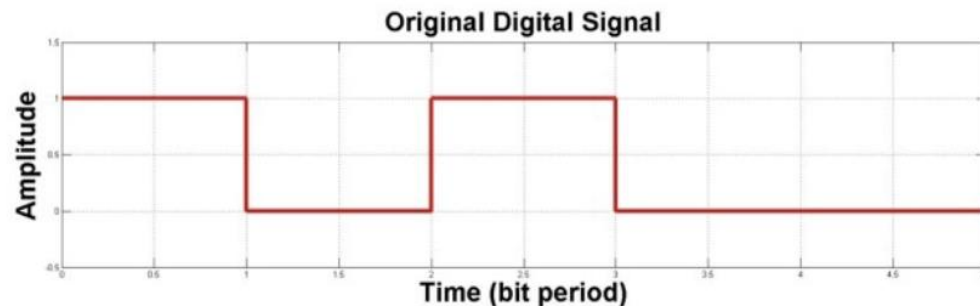
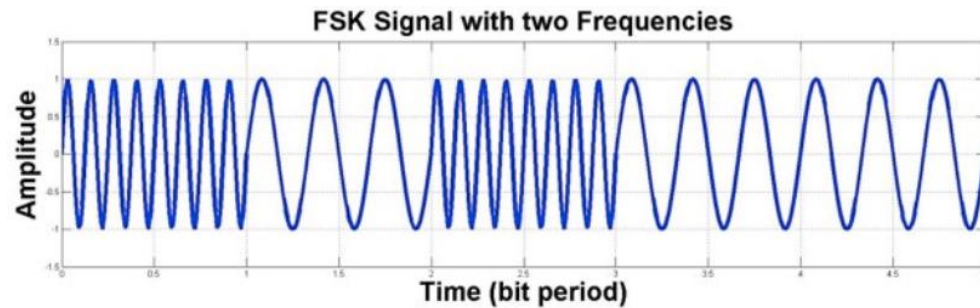
Why doesn't BLE avoid WiFi altogether?

Can't on 2.4 GHz

Wants 2.4 GHz for technology improvements

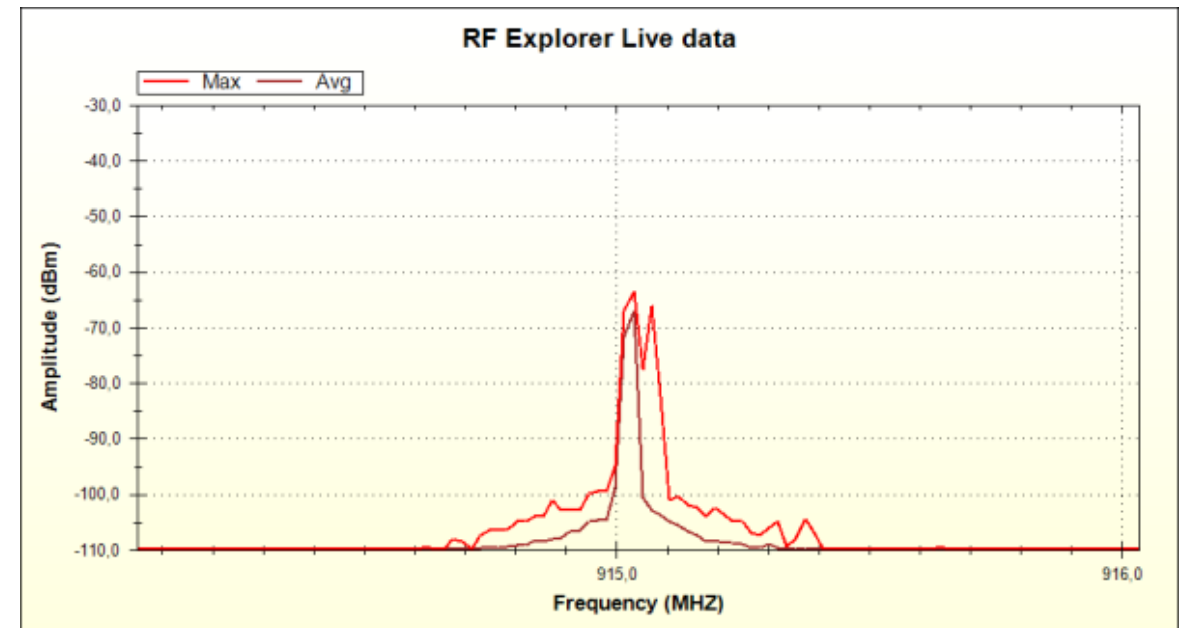
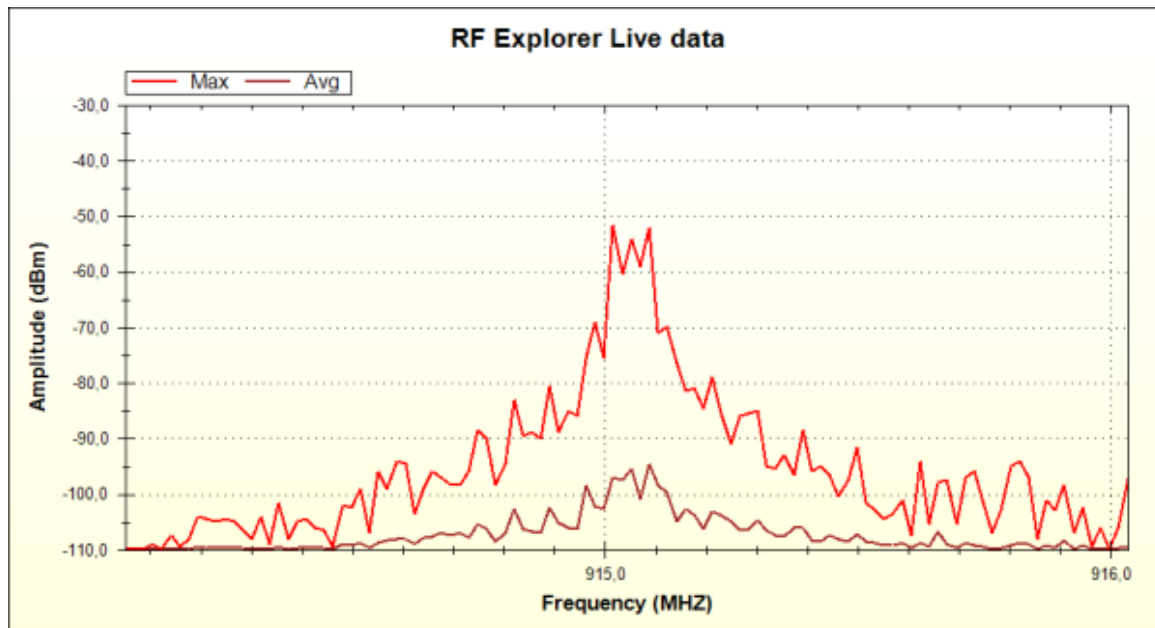
BLE modulation

- Gaussian Frequency-Shift Keying (GFSK)
 - Improvement on base Frequency-shift Keying
 - Smoother transitions between bits -> reduces nearby interference



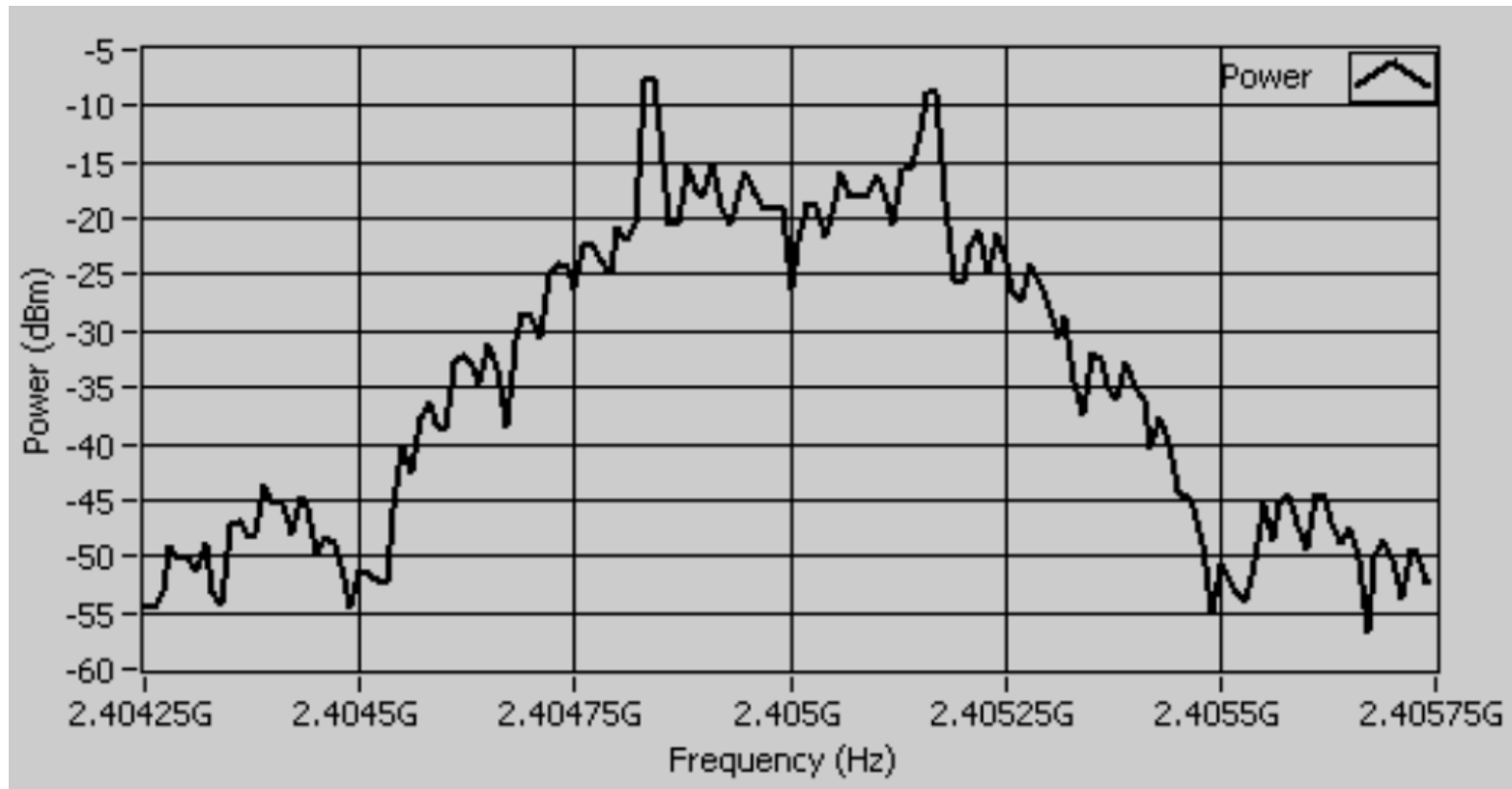
Why use GFSK?

- Gaussian FSK lessens spectral leakage at the expense of some loss in intersymbol discriminability
 - Translation: GFSK reduces bandwidth at the cost of bit errors



An example from `good case` BLE hardware

- 2 MHz BLE channels
 - Energy focused on ± 0.55 MHz
 - With some much smaller energy near the edges of the channel



BLE signal strength

The requirements for a Bluetooth low energy radio are as follows:

Feature	Value
Minimum TX power	0.01 mW (-20 dBm)
Maximum TX power	100 mW (20 dBm)
Minimum RX sensitivity	-70 dBm (BER 0.1%)

The typical range for Bluetooth low energy radios is as follows:

TX power	RX sensitivity	Antenna gain	Range
0 dBm	-92 dBm	-5 dB	160 meters
10 dBm	-92 dBm	-5 dB	295 meters

The range to a smart phone is typically 0-50 meters due to limited RF performance of the phones.

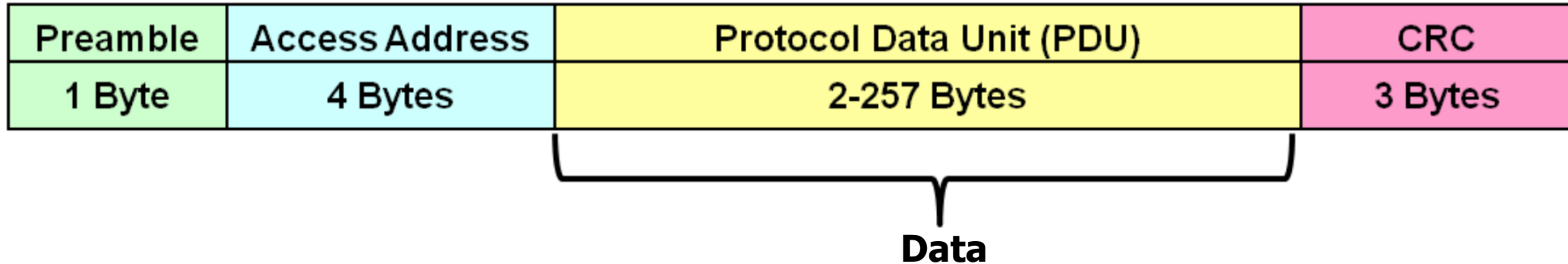
- Remember nRF52840 capabilities
 - Transmit: up to 8 dBm
 - Receive sensitivity: -95 dBm

Outline

- **BLE Layers**
 - Physical Layer
 - **Link Layer**
- BLE roles
 - Advertising
 - Scanning
 - Scan Responses
- Communicating with advertisements
 - Advertisement Use Cases
 - Energy Use
 - Packet Collisions

Packet structure

BLE Packet



- Same packet structure for both advertisements and connections
 - Fields are filled in little endian (the opposite of network byte order 😡)
- Access address unique for each connection
 - Somewhat randomly chosen (spec has rules)
 - In Advertising always set to 0x8E89BED6

Device addresses

- Public and private address forms
- Public
 - 48 bits: 24-bits of company ID, 24-bits of company assigned number
 - Literally the same MAC address scheme as Ethernet and WiFi
- Private
 - Top two MSbs specify type. Options:
 - 46 bits of random
 - 46 bits of hash of an identity key
- **Why have the two types?**

Device addresses

- Public and private address forms
- Public
 - 48 bits: 24-bits of company ID, 24-bits of company assigned number
 - Literally the same MAC address scheme as Ethernet and WiFi
- Private
 - Top two MSbs specify type. Options:
 - 46 bits of random
 - 46 bits of hash of an identity key
- **Why have the two types? Privacy**

Data whitening

- Avoid long series of repetitive bits (all zeros or all ones)
 - Would cause RF noise to be more focused in one direction
 - Radio hardware desires output to have zero DC-bias (or close to that)
 - Great example of the PHY and MAC being interwoven in wireless

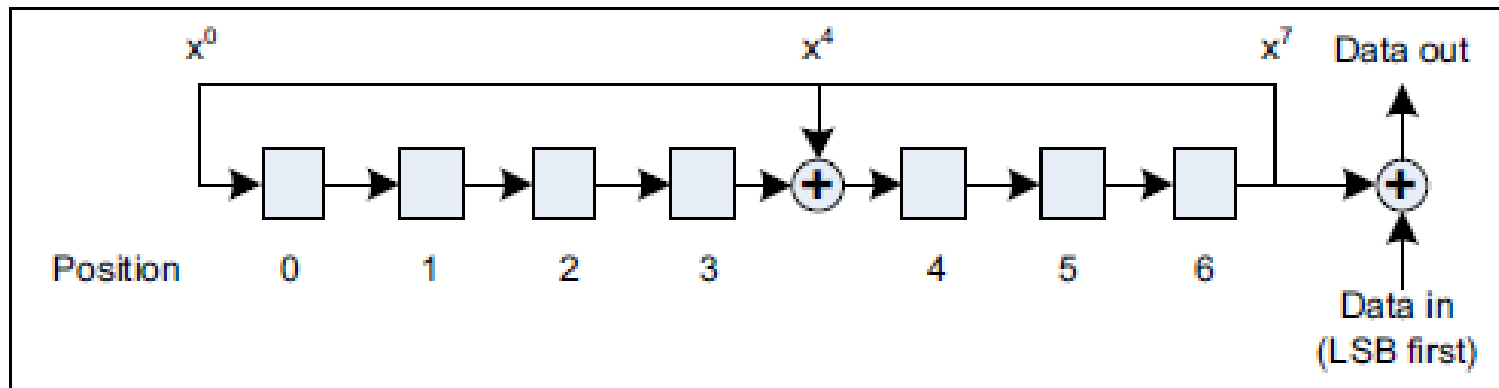


Figure 3.3: The LFSR circuit to generate data whitening

- I always forget this exists, since hardware usually handles it automatically

Bit processing pipeline

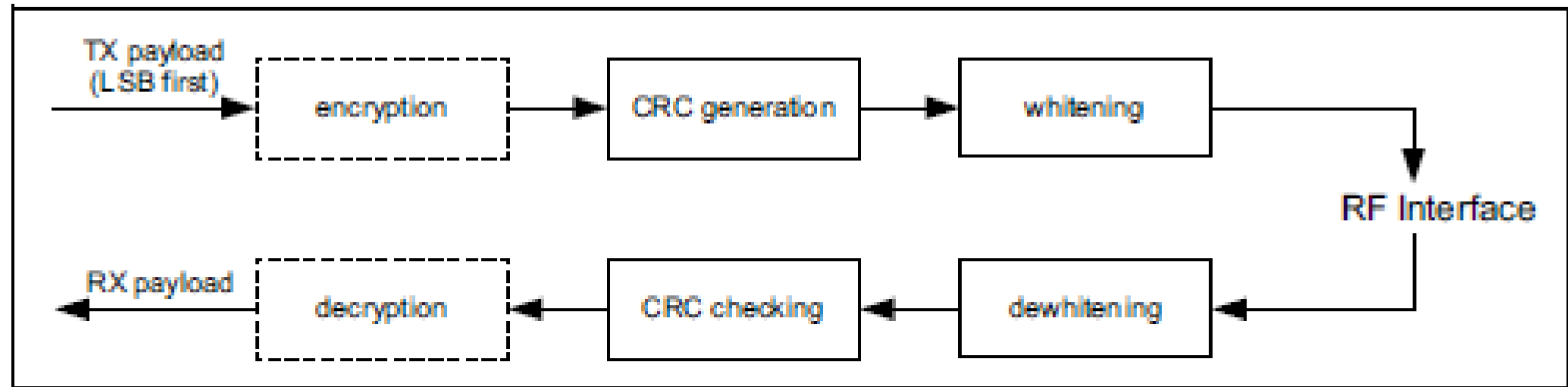


Figure 3.1: Payload bit processes for the LE Uncoded PHYs

Break + Question

- With enough scanners, could you track BLE devices as they move?

Break + Question

- With enough scanners, could you track BLE devices as they move?
 - Link Layer
 - Depends on how long they use a device address for
 - You can do a scan of BLE transmissions to find device addresses
 - Scans at multiple locations can detect when a device moves throughout an area
 - But if the device re-randomizes between two scanners, you can't follow it anymore
 - Re-randomizing at a scanner could be detectable...
 - Or if the user has more than one device with unsynchronized rotation schedules

Break + Question

- With enough scanners, could you track BLE devices as they move?
- Physical Layer
 - Fingerprint unique physical-layer imperfections in signals
 - Looking at things like amplitude and timing
 - 2022 paper out of UCSD explores this

Evaluating Physical-Layer BLE Location Tracking Attacks on Mobile Devices

Hadi Givvehchian*, Nishant Bhaskar*, Eliana Rodriguez Herrera, Héctor Rodrigo López Soto, Christian Dameff, Dinesh Bharadia, and Aaron Schulman

UC San Diego

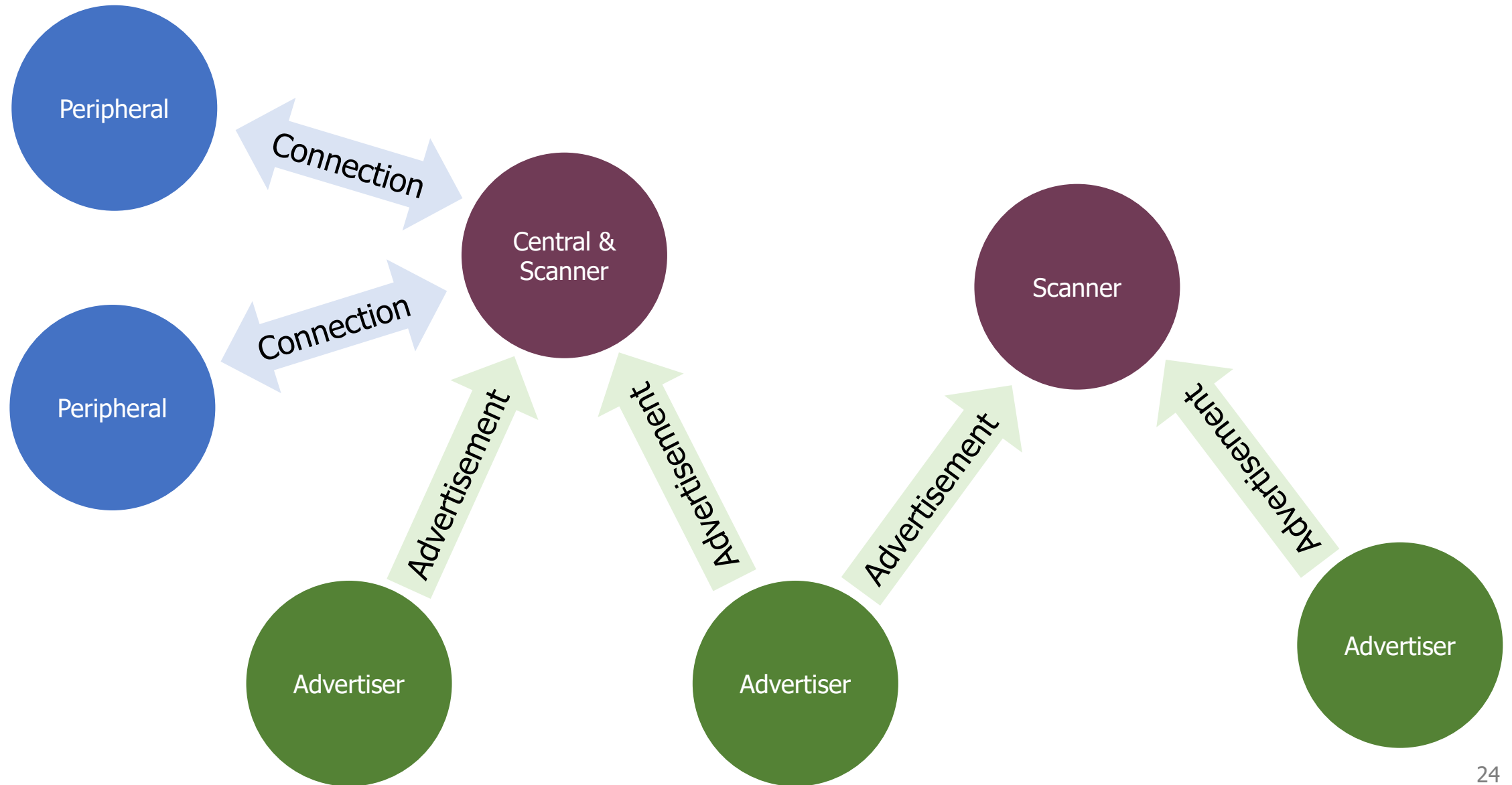
Abstract—Mobile devices increasingly function as wireless tracking beacons. Using the Bluetooth Low Energy (BLE) protocol, mobile devices such as smartphones and smartwatches continuously transmit beacons to inform passive listeners about device locations for applications such as digital contact tracing for COVID-19, and even finding lost devices. These applications

countermeasures by fingerprinting the device at a lower layer. Specifically, prior work has demonstrated that wireless transmitters have imperfections introduced in manufacturing that produce a unique physical-layer fingerprint for that device (e.g., Carrier Frequency Offset and I/Q Offset). Physical-layer

Outline

- BLE Layers
 - Physical Layer
 - Link Layer
- **BLE roles**
 - **Advertising**
 - Scanning
 - Scan Responses
- Communicating with advertisements
 - Advertisement Use Cases
 - Energy Use
 - Packet Collisions

BLE network topology



Advertising

- BLE discovery mechanism
 - Make nearby devices aware of advertiser's existence
 - Communicate some information from or about advertiser
 - Traditional purpose is to enable connections, but this is also useful for general communication
- Advertisements
 - Periodic broadcast messages with data
- Scan Requests/Responses
 - Scanner sends responses after getting a request
 - Only occurs when scanner is listening
 - Almost literally "bonus advertisement data"

Advertising packet layering

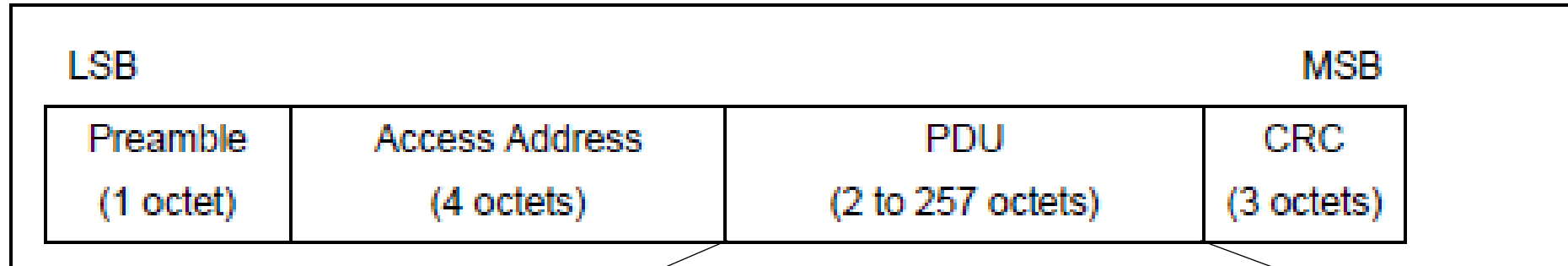


Figure 2.1: Link Layer packet format

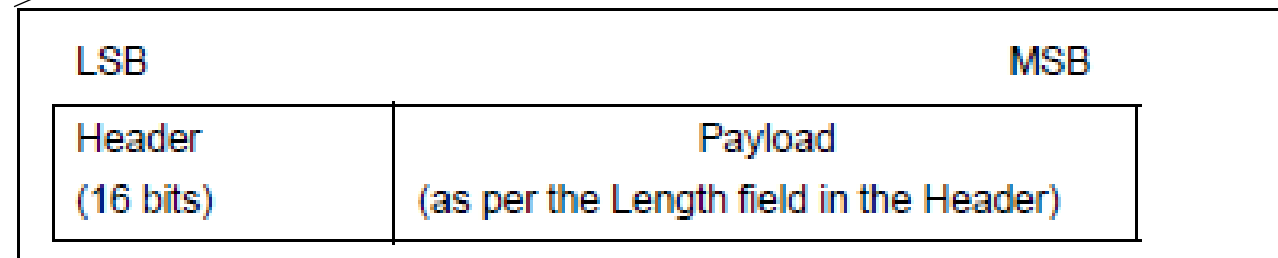


Figure 2.2: Advertising channel PDU

BLE advertising header

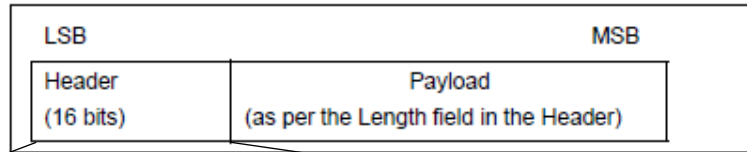


Figure 2.2: Advertising channel PDU

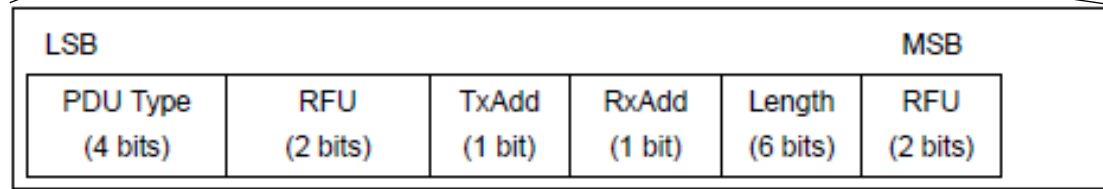


Figure 2.3: Advertising channel PDU Header

PDU Type $b_3b_2b_1b_0$	Packet Name
0000	ADV_IND
0001	ADV_DIRECT_IND
0010	ADV_NONCONN_IND
0011	SCAN_REQ
0100	SCAN_RSP
0101	CONNECT_REQ
0110	ADV_SCAN_IND
0111-1111	Reserved

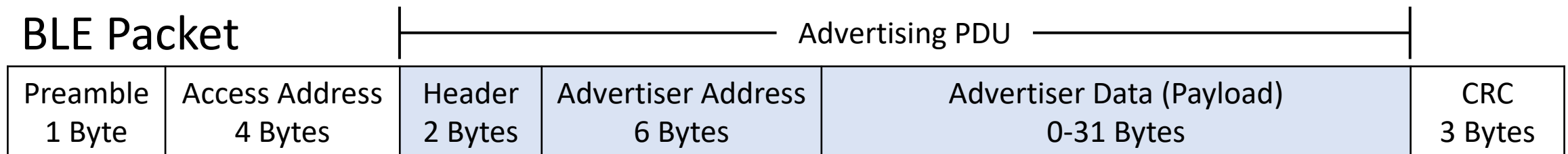
Table 2.1: Advertising channel PDU Header's PDU Type field encoding

- ADV_IND
 - Advertisement
 - Allows connections and scan requests
- ADV_NONCONN_IND
 - Advertisement
 - No connections or scan requests
- ADV_SCAN_IND
 - Advertisement
 - No connections but allows scan requests
- SCAN_REQ
 - Scan request
- SCAN_RSP
 - Scan response

Advertisement payloads

Payload	
AdvA (6 octets)	AdvData (0-31 octets)

- AdvA – address of the advertiser
 - TxAdd bit from header specifies if this is a “public” or “random” address
- Remaining up to 31 bytes are available for use
- Putting it all together, up to 47 bytes total:



Scan Requests and Responses

- Scan request
 - Request for additional information
 - Just the two addresses: the scanner's and the advertiser's
- Scan response
 - Identical to an advertisement in structure but only occurs after a request
 - Usually has additional data not in the advertisement

Payload	
ScanA (6 octets)	AdvA (6 octets)

Figure 2.8: SCAN_REQ PDU Payload

Payload	
AdvA (6 octets)	ScanRspData (0-31 octets)

Figure 2.9: SCAN_RSP PDU payload

Advertising timing



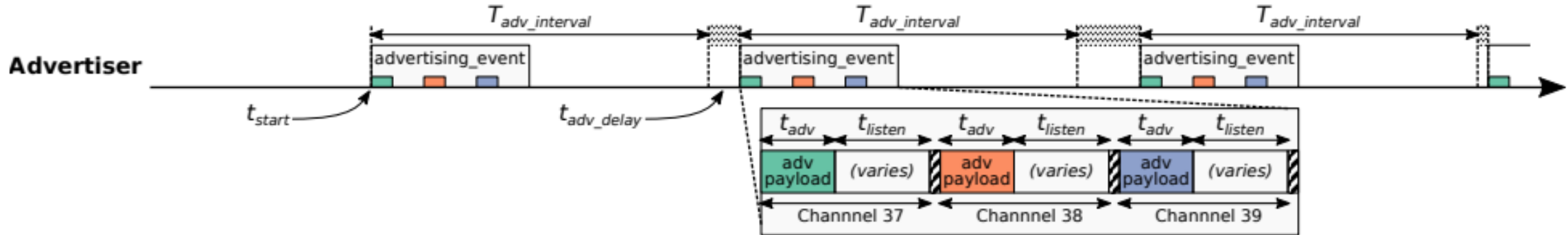
- Advertising Events occur periodically [20ms – 10.24 s] (or longer)
 - Plus a random delay after each instance [0-10ms]
 - **Why?**
- User picks the rate as a tradeoff of energy and discovery latency

Advertising timing



- Advertising Events occur periodically [20ms – 10.24 s] (or longer)
 - Plus a random delay after each instance [0-10ms]
 - **Why? Avoid repeat collisions**
- User picks the rate as a tradeoff of energy and discovery latency

Advertising event



- Three transmissions, one on each advertising channel
 - Always in the same order
- Transmission, followed by listening window on that same channel
 - Requests will be sent ≥ 150 us (Inter-Frame Spacing, IFS) after Tx
 - Followed by a retune to the next channel frequency
- This short listen window is the magic “low energy” part

Preserving energy in communication

- Most energy is spent listening
 - This is due primarily to how long listening durations are compared to transmissions
- Example: maximum-sized BLE transmission:
 - $8 \text{ bits/byte} * 47 \text{ bytes} = 376 \text{ bits}$ at 1 Mbps = 0.376 ms transmitting
 - So listening for an entire second is >2500 times longer

Payload of an advertisement

- What do you stick in the BLE payload anyways?
 - Theoretically whatever you want, but that isn't very compatible
 - Point is to specify capabilities of the advertiser
- Desire: specify payloads in such a way that all scanners can interpret what they mean about the device
 - This is different from traditional internet packets
 - Broadcasts are for anyone to hear, not a specific server/application
- Which fields are or aren't present is device-specific
 - A lot more possible fields than will really be used on any device

TLV Format

- Type – Length – Value ([Wikipedia](#))
 - Actually, BLE does the length part first
 - Scanner can hop through length/type pairs to find what interests it

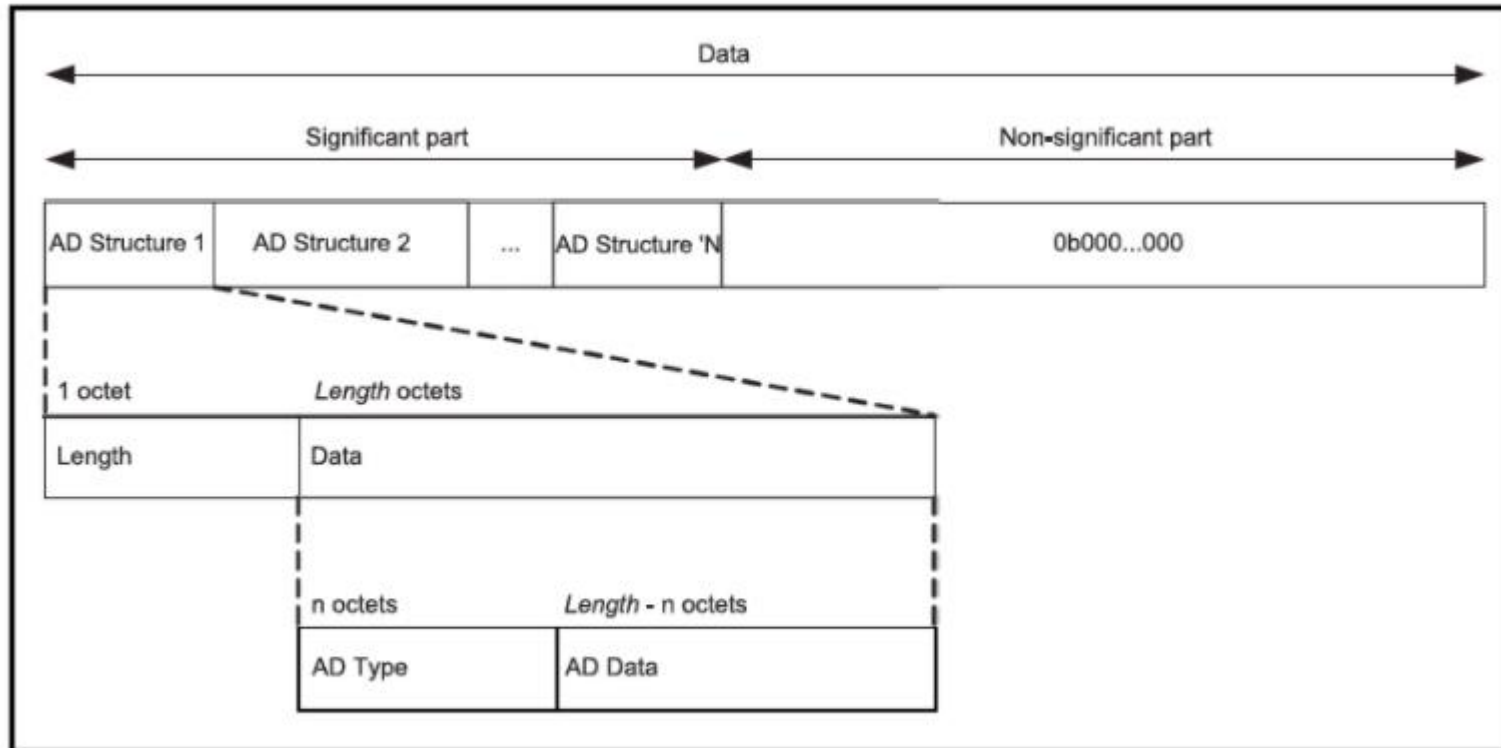


Figure 11.1: Advertising and Scan Response data format

Payload types

- Listed in the Core Specification Supplement [[Supplement v9](#)]
 - Each might have their own considerations about AD Data format
- Flags (supported modes: BLE and Bluetooth) required by Apple?
- Name
- Service UUID
- TX Power Level
- Manufacturer-specific data
- And about twenty others

Example: BLE advertisement payload

This particular payload (52 bytes) is a combination of advertisement data and scan response data

- Example payload:

02	01	06	02	0A	13	11	07	BA	DC	7C	89	24	45	F2	84
12	38	85	32	01	00	42	7A	0F	FF	4E	03	00	00	00	00
00	00	00	00	00	00	00	00	0B	09	53	41	42	52	45	2D
46	37	43	31												

Details:

LEN.	TYPE	VALUE
2	0x01	0x06
2	0x0A	0x13
17	0x07	0xBADC7C892445F284123885320100427A
15	0xFF	0x4E030000000000000000000000000000
11	0x09	0x53414252452D46374331

Break + Question:

What is the Local Name of this device?

Type 0x09

Bytes in the Local Name type are ASCII values

Bonus: what is this device?

Example: BLE advertisement payload


This particular payload (52 bytes) is a combination of advertisement data and scan response data

- Example payload:

02	01	06	02	0A	13	11	07	BA	DC	7C	89	24	45	F2	84
12	38	85	32	01	00	42	7A	0F	FF	4E	03	00	00	00	00
00	00	00	00	00	00	00	00	0B	09	53	41	42	52	45	2D
46	37	43	31												

Details:

LEN.	TYPE	VALUE
2	0x01	0x06
2	0x0A	0x13
17	0x07	0xBADC7C892445F284123885320100427A
15	0xFF	0x4E030000000000000000000000000000
11	0x09	0x53414252452D46374331

 **SABRE-F7C1** CONNECT

0C:43:14:2C:F7:C1

NOT BONDED ▲ -67 dBm ↔ 108 ms

Device type: LE only

Advertising type: Legacy

Flags: LE General Discoverable, BR\EDR Not Supported

Tx Power Level: 19 dBm

Complete list of 128-bit Service UUIDs:
7a420001-3285-3812-84f2-4524897cdcba

Manufacturer data (Bluetooth Core 4.1):
Company: SafeTrust Inc. <0x034E>
0x00000000000000000000000000000000

Complete Local Name: SABRE-F7C1

What is this device?

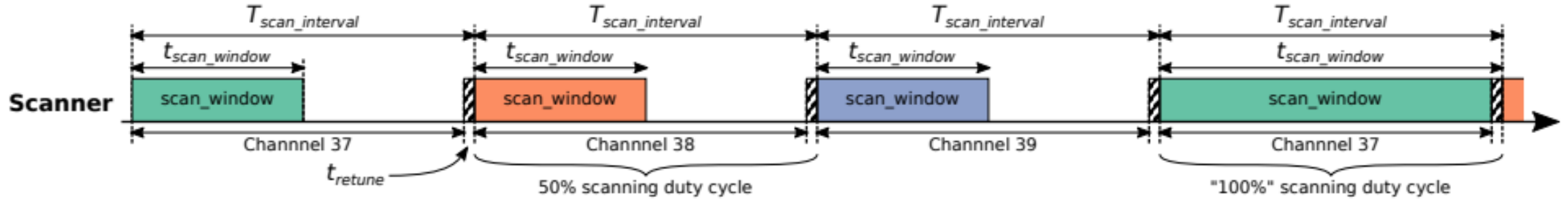
- Bluetooth module for door ID scanner



Outline

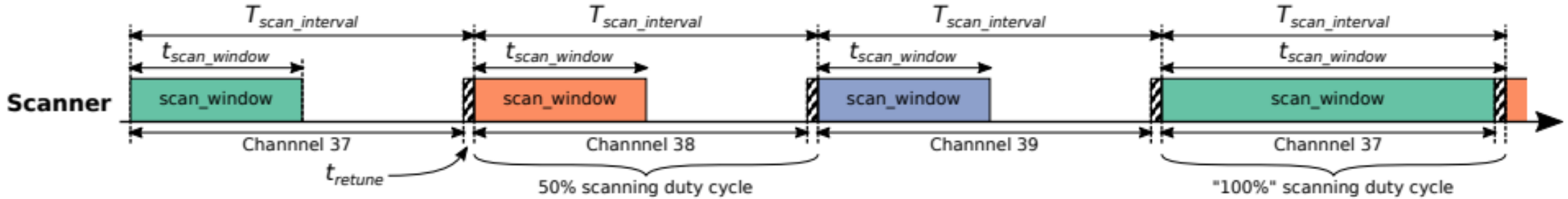
- BLE Layers
 - Physical Layer
 - Link Layer
- **BLE roles**
 - Advertising
 - **Scanning**
 - Scan Responses
- Communicating with advertisements
 - Advertisement Use Cases
 - Energy Use
 - Packet Collisions

Scanning Pattern



- Iterate through channels, listening for advertisements
 - $T_{scan_interval}$ controls rate at which channels are changed
 - T_{scan_window} controls duty cycle of listening
- **Why listen at a low duty cycle?**

Scanning Pattern



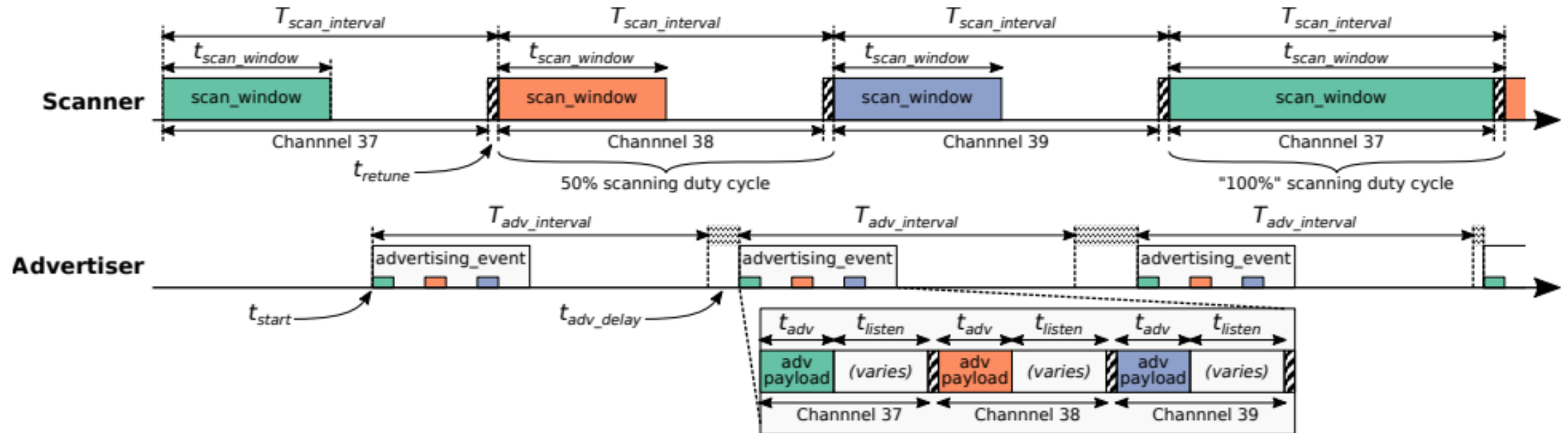
- Iterate through channels, listening for advertisements
 - $T_{scan_interval}$ controls rate at which channels are changes
 - T_{scan_window} controls duty cycle of listening
- **Why listen at a low duty cycle? Save energy**

A warning about scanning expectations

- Scanners will NOT receive 100% of packets sent
 - Even ignoring range issues
- Packets are lost due to (in roughly descending order):
 - Duty cycle
 - Sharing 2.4 GHz antenna with WiFi
 - Retune period after each scanning interval
 - Dropped packets in the receive software
 - Packet collisions

Putting it all together

- Advertisements are received when the channel of the scan window and the channel of the advertisement overlap in time (and space)



Outline

- BLE Layers
 - Physical Layer
 - Link Layer
- **BLE roles**
 - Advertising
 - Scanning
 - **Scan Responses**
- Communicating with advertisements
 - Advertisement Use Cases
 - Energy Use
 - Packet Collisions

Scan requests/responses seem intriguing

- Why not send most data in scan responses instead of advertisements?
 - Theoretically could reduce energy costs
- Scan we use scan requests as a form of acknowledgement?
 - Could relieve need for redundant transmissions
- Problem: scan requests/responses don't work all that well

Scan Requests and Responses are broken

- Goal: provide a little extra advertisement data on demand
- Problem: exponential backoff for lost messages
 - Scanners backoff if there is a request without a response
 - Assumption: collision with another scanner
 - But collisions also occur with other devices advertising
 - Which *shouldn't* require backoff
 - Result: scan requests occur less frequently than expected/desired
 - Instead, just send additional data in additional advertisements

Kravets, Robin, Albert F. Harris III, and Roy Want. "Beacon trains: blazing a trail through dense BLE environments." *Proceedings of the Eleventh ACM Workshop on Challenged Networks*. 2016.

Outline

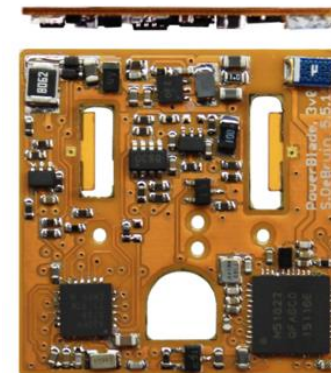
- BLE Layers
 - Physical Layer
 - Link Layer
- BLE roles
 - Advertising
 - Scanning
 - Scan Responses
- **Communicating with advertisements**
 - **Advertisement Use Cases**
 - Energy Use
 - Packet Collisions

Advertisements are sufficient for some applications

BLE advertisements are uncoordinated,
broadcast messages designed for discovery.

Devices are being deployed using
advertisements.

1. Beacons – iBeacon
2. Tracking – Tile
3. Local communication – Apple Continuity
4. Sensor deployments – PowerBlade



Beacons

- Advertising with advertisements!
- Web of Things
 - Real-world tags that broadcast virtual-world identifiers
- iBeacon and Eddystone
 - Formats for sending URLs and device identifiers
 - Use existing BLE fields (Service Data and Manufacturer-Specific Data)
- Faded in popularity significantly



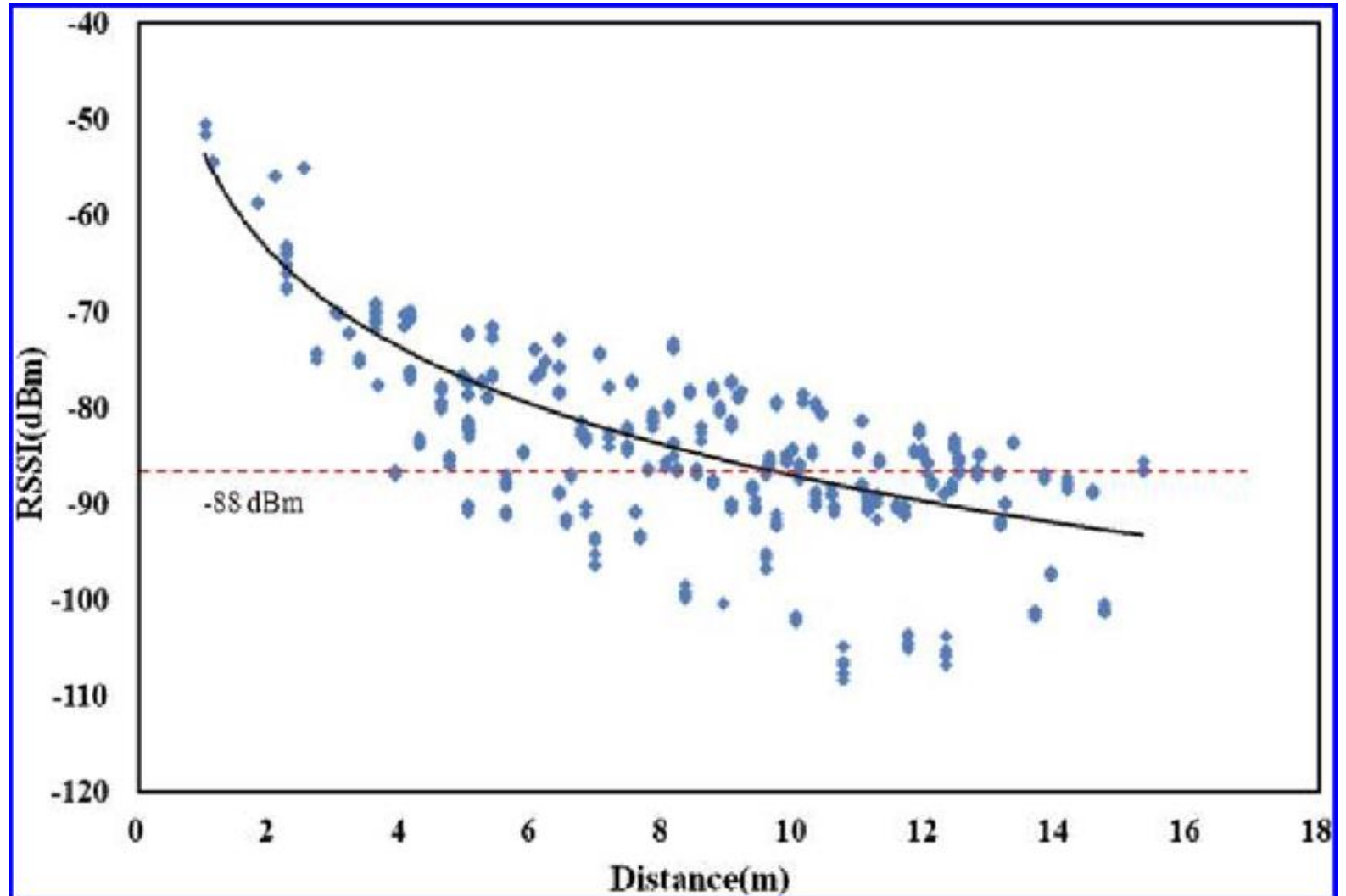
Tracking

- Find devices nearby
 - Get a sense of distance to the device
- Find my X
 - Tile: find my keys
 - Apple: find my device (before UWB radios)
- Uses TX power level field
 - Lists the transmitted power of the device
 - Pathloss = TX power – RSSI (all in dBm)



Problem with RSSI-based distance – not accurate

- Pathloss is NOT only due to distance
- RSSI is way worse at this than you hope it would be



Citation: literally everyone has made this figure at some point

Local communication: Apple “Continuity”

- Key part of Apple’s “ecosystem”, including:
- Handoff
 - Start tasks on one device and continue on another device
- Universal Clipboard,
 - Copying of data from one Apple device and pasting on another
- iPhone Cellular Calls
 - Make calls using iPhone’s cellular connection on Mac or iPad



Communication with only *nearby* devices

- Uses BLE “Manufacturer Specific Data” to communicate



Table 1. Advertisement Frames

Test 1 Test 2			
		Count	
Address Type	Public Random	26 726	57 1,518
Company ID†	Apple	692	1296
	Microsoft	30	201
	Garmin	2	9
	Samsung	0	3
	All Others	2	9
† Randomized Devices Only			

0	7	8	15	16	23	24	31
Access Address - 0x8E89BED6							
Packet Header							
Advertising Address - xx:xx:xx:xx:xx:xx							
Length / Type - 0x01 / Flags (Optional)						Length	
Type - 0xFF		Company ID - 0x004C				Apple Type	
Apple Length		Variable Length Apple Data				Apple Type	
Apple Length		Variable Length Apple Data					

Type	Value
Watch Connection	11
Handoff	12
Wi-Fi Settings	13
Instant Hotspot	14
Wi-Fi Join Network	15
Nearby	16

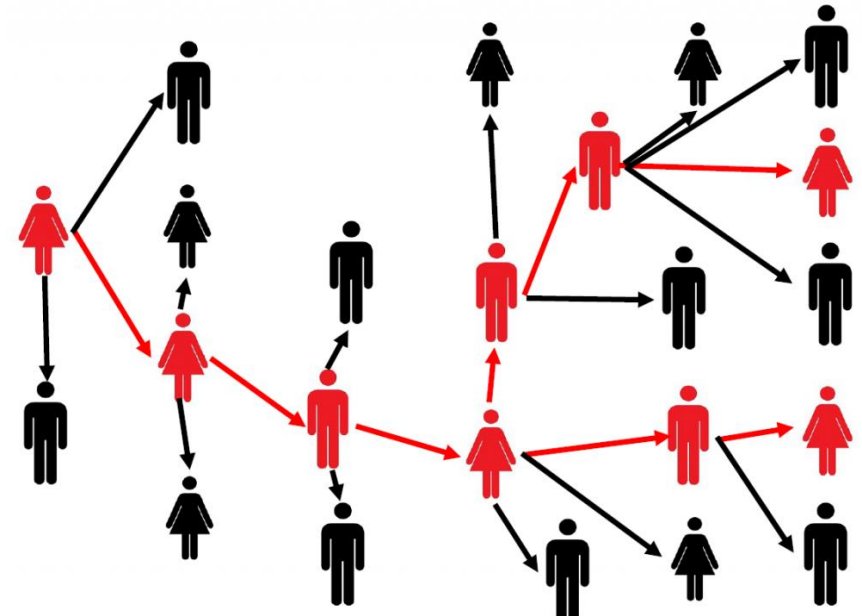
Table 3. Action Codes

Type	Description
1	iOS recently updated
3	Locked Screen
7	Transition Phase
10	Locked Screen, Inform Apple Watch
11	Active User
13	Unknown
14	Phone Call or Facetime

Martin, Jeremy, et al. "Handoff all your privacy—a review of apple's bluetooth low energy continuity protocol." *Proceedings on Privacy Enhancing Technologies* 2019.4 (2019): 34-53.

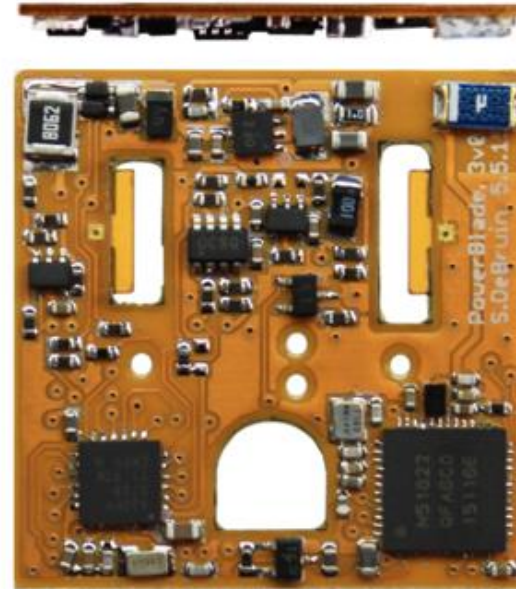
Local Communication: Exposure Notifications

- Apple and Google collaboration to use phones for contact tracing
 - Smartphone constantly broadcasts identifier.
 - Periodically, each smartphone listens for broadcasts around it.
 - Check list of identifiers to see if you've been around someone who is sick.
- Requires government/healthcare system interactions to determine when an identifier should be flagged as sick
 - 24 states (not Illinois) adopted this
- Implemented at OS level in background



Sensor deployments

- Report data so gateways and users can retrieve it simultaneously
 - Easy introspection during deployment
 - Satisfy users' curiosity
- Ignore difficult questions about networking
 - Just broadcast the data!



DeBruin, Samuel, et al. "Powerblade: A low-profile, true-power, plug-through energy meter." *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. 2015.

Outline

- BLE Layers
 - Physical Layer
 - Link Layer
- BLE roles
 - Advertising
 - Scanning
 - Scan Responses
- **Communicating with advertisements**
 - Advertisement Use Cases
 - **Energy Use**
 - Packet Collisions

Paper: power measurements of BLE advertisements

Schrader, Raphael, et al. "Advertising power consumption of bluetooth low energy systems." *2016 3rd International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS)*. IEEE, 2016.

The 3rd IEEE International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems
26-27 September 2016, Offenburg, Germany

Advertising Power Consumption of Bluetooth Low Energy Systems

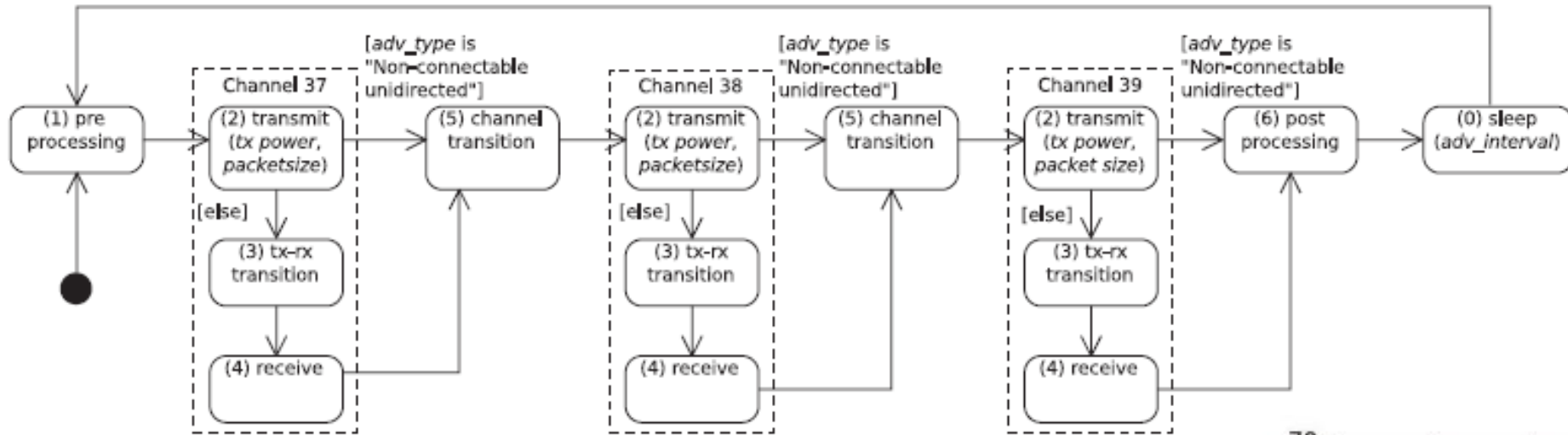
Raphael Schrader, Thomas Ax, Christof Röhrig, Claus Fühner

Fachhochschule Dortmund

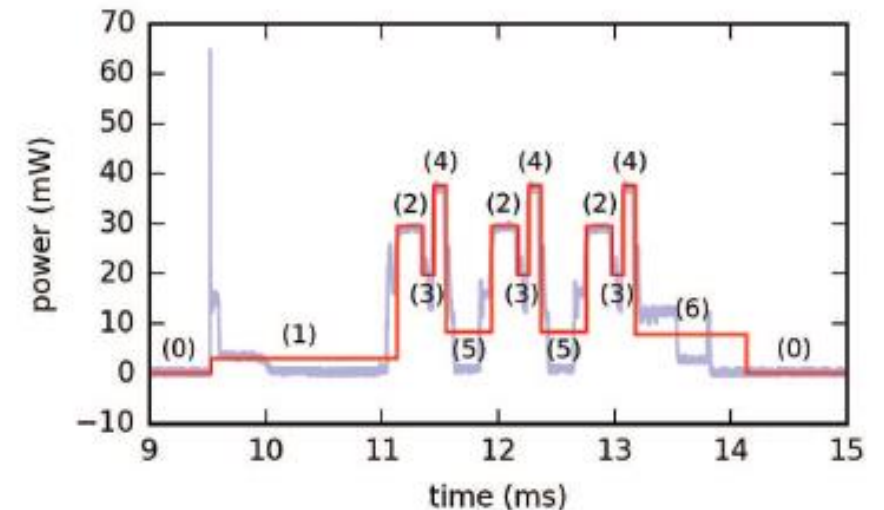
Fachbereich Informatik

Email: claus.fuehner@fh-dortmund.de

Energy model for BLE advertisements



- Creates a set of states and metrics for how much power each uses and for what duration
 - $\text{power} * \text{duration} = \text{energy}$



Measurements of Power Use

- Power use and duration (energy)

- nRF51 (nRF51822)
- nRF52 (nRF52832)

TABLE II
SOC-DEPENDENT MODEL PARAMETERS FROM MEASUREMENTS

Phase	Nordic nRF51		Nordic nRF52	
	T_i (σ) (μ s)	P_i (mW)	T_i (σ) (μ s)	P_i (mW)
preprocessing	951.8 (9.1)	2.9	321.4 (8.9)	2.7
tx (4 dBm)	72.4 (0.5) + $n_{\text{Bit}} \cdot 1/\text{Bit}$	45.4	13.2 (1.8) + $n_{\text{Bit}} \cdot 1/\text{Bit}$	46.2
tx (0 dBm)		29.5		33.2
tx (-4 dBm)		25.8		27.5
tx (-8 dBm)		23.2		25.3
tx (-12 dBm)		21.1		23.6
tx (-16 dBm)		19.8		22.6
tx (-20 dBm)		18.9		21.6
tx-rx transit.	94.7 (0.6)	19.6	130.6 (2.0)	15.9
rx	104.3 (1.5)	37.6	73.0 (3.9)	32.4
channel transit.	390.4 (0.9)	8.4	432.3 (4.47)	7.3
postprocessing	961.8 (156.9)	7.7	321.4 (32.2)	10.2
sleep	T_{advSleep}	0.0114	T_{advSleep}	0.0058

How much energy does it cost to send data over advertisements?

- Configuration

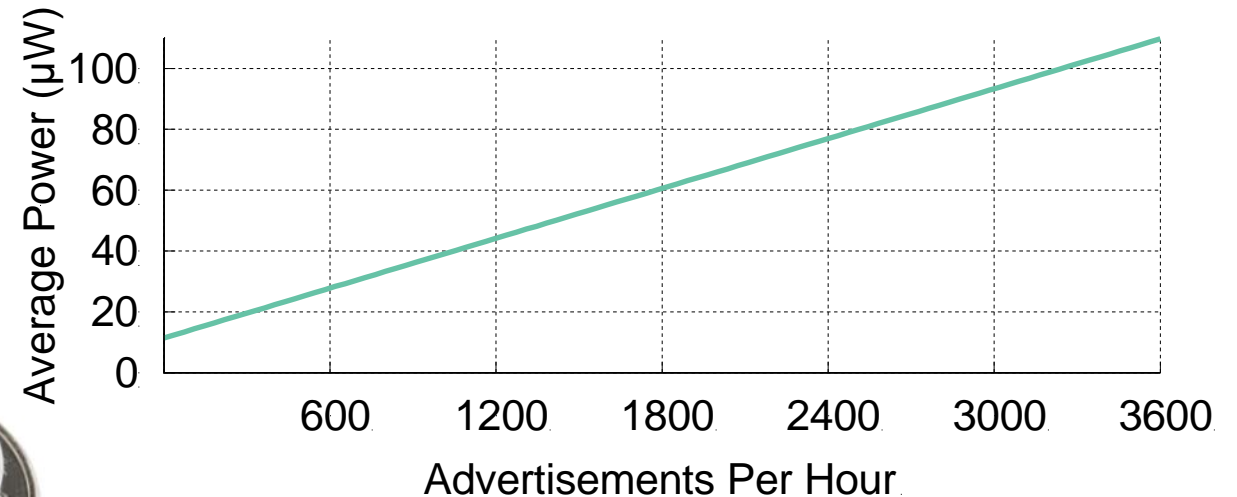
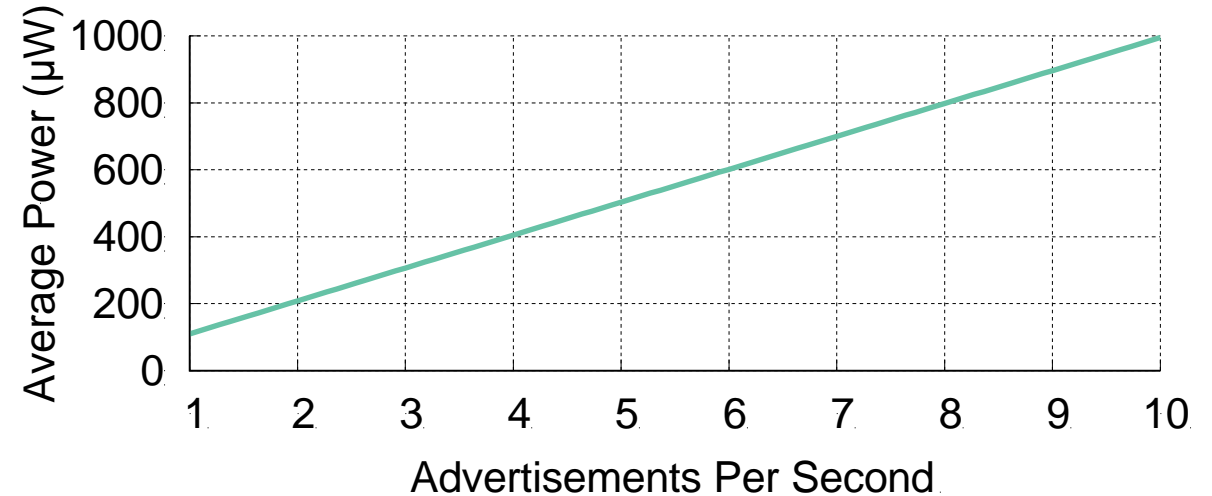
- nRF51822 microcontroller
- Maximum payload size
- +4 dBm transmit power
- Connectable advertisement
- Sleep power 11 μW

- One packet per second example:

- 110 μW average
- ~270 days on a CR2032

- One packet per minute example:

- 13 μW average
- ~2250 days on a CR2032



Break + Open Question

- How long does the lifetime of a BLE advertiser need to be?

Outline

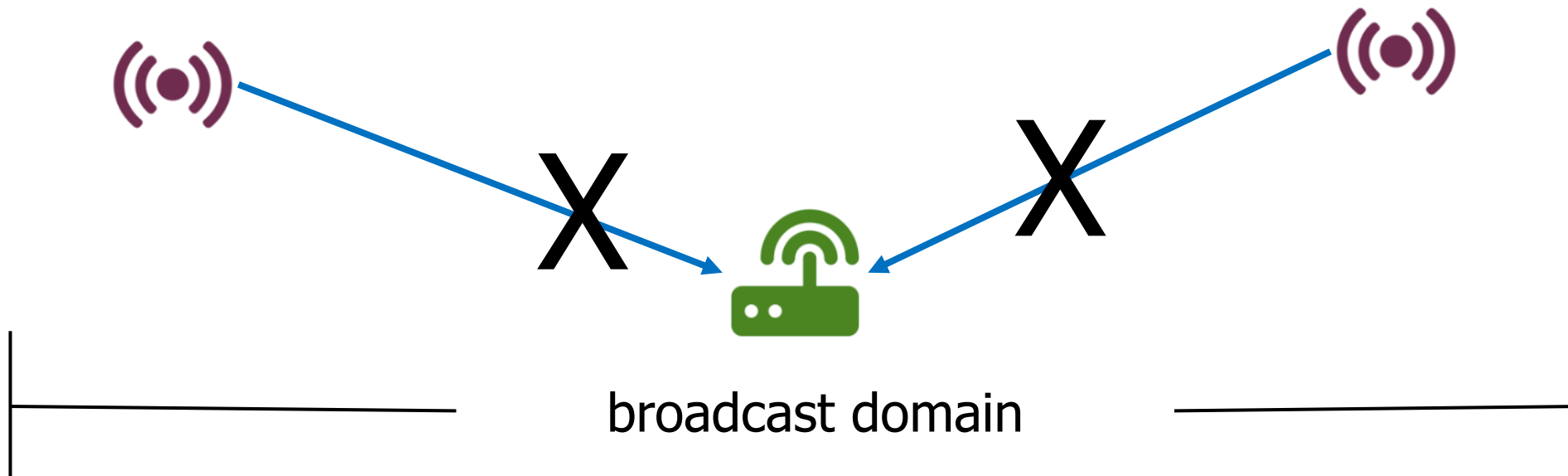
- BLE Layers
 - Physical Layer
 - Link Layer
- BLE roles
 - Advertising
 - Scanning
 - Scan Responses
- **Communicating with advertisements**
 - Advertisement Use Cases
 - Energy Use
 - **Packet Collisions**

Questions about network capability

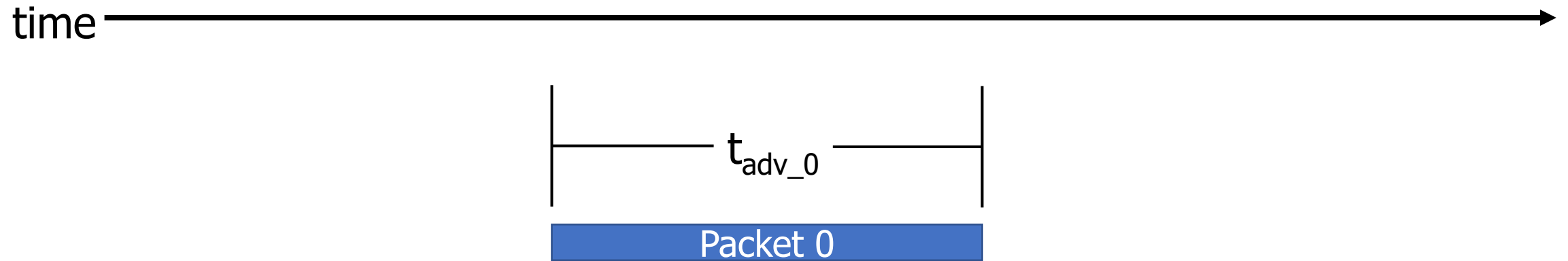
- What are the odds that a transmitted advertisement will be received?
 - Packet reception rate
- If M redundant advertisements are sent instead, what are the odds that at least one are received?
 - Data reception rate
- How do these odds vary with number of devices, advertising interval, and packet size?

What causes transmissions not to be received?

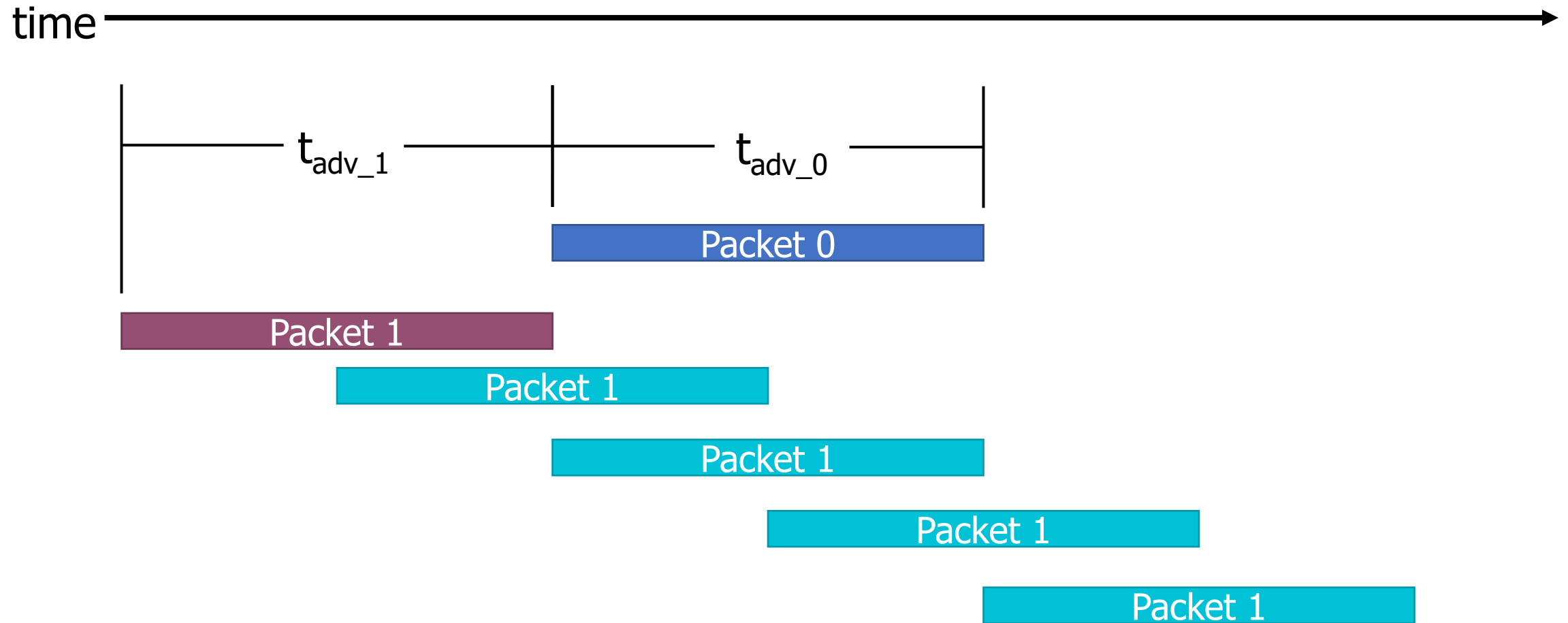
1. Not within range of the gateway.
 - Or various other losses within the gateway itself
2. Two devices try to send at the same time (packet collision).



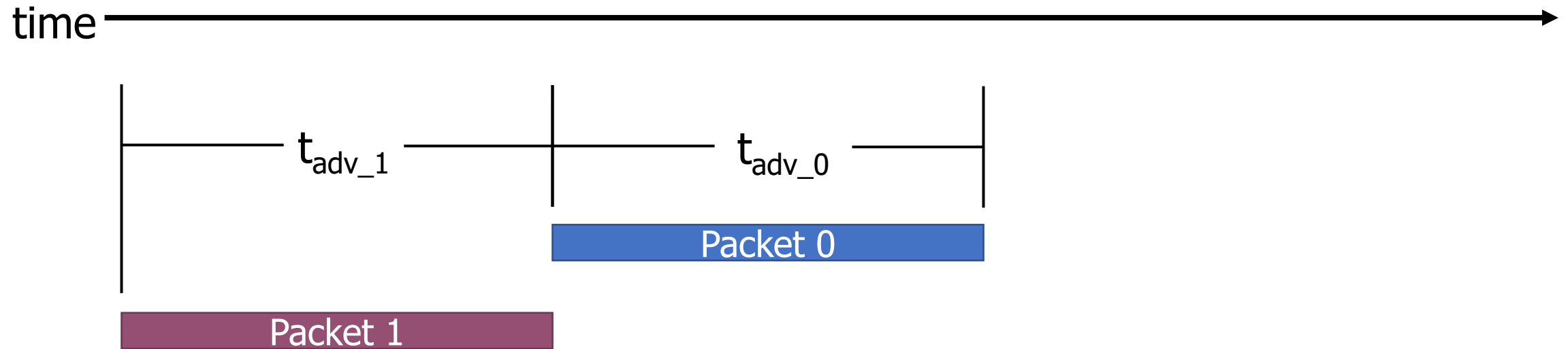
What is the probability of a packet collision?



What is the probability of a packet collision?



What is the probability of a packet collision?



$$\text{Probability of Collision} = \frac{\text{Vulnerable Period}}{\text{Transmission Window}} = \frac{t_{adv_1} + t_{adv_0}}{T_{adv_interval} + E(t_{adv_delay})}$$

Determine Probability of Multiple Failures

- Given:
 - Probability of Collision
- Determine:
 - Probability of Reception for data sent redundantly across **M** packets
 - i.e., what are the odds that **at least one** of the packets doesn't collide
 - $1 - (\text{Probability of Collision})^M$
 - $(P_c)^M$ = Probability that all of them collide
 - $1 - \text{that}$ = Probability that NOT all of them collide

How do we determine reception rate?

With redundancy, we care about data reception instead of packet reception.

Naïve model:

- *Packet Reception Rate* = $1 - (\text{Probability of Collision})$
- *Data Reception Rate* = $1 - (\text{Probability of Collision})^{\text{Number of Packets}}$

Data Reception Assumption: repeat packet collisions are independent.

- True for any arbitrary selection of two BLE devices
- False for two devices that have recently collided (but difference is $\sim 1\%$)

Equations for modeling data transmissions

- Packet Reception Rate

- Probability that at the transmitted packet does not have a collision with any of N transmitting devices

$$PRR = \left(1 - \frac{2 * t_{adv}}{T_{adv_interval} + E[t_{adv_delay}]}\right)^{N-1}$$

- Data Reception Rate

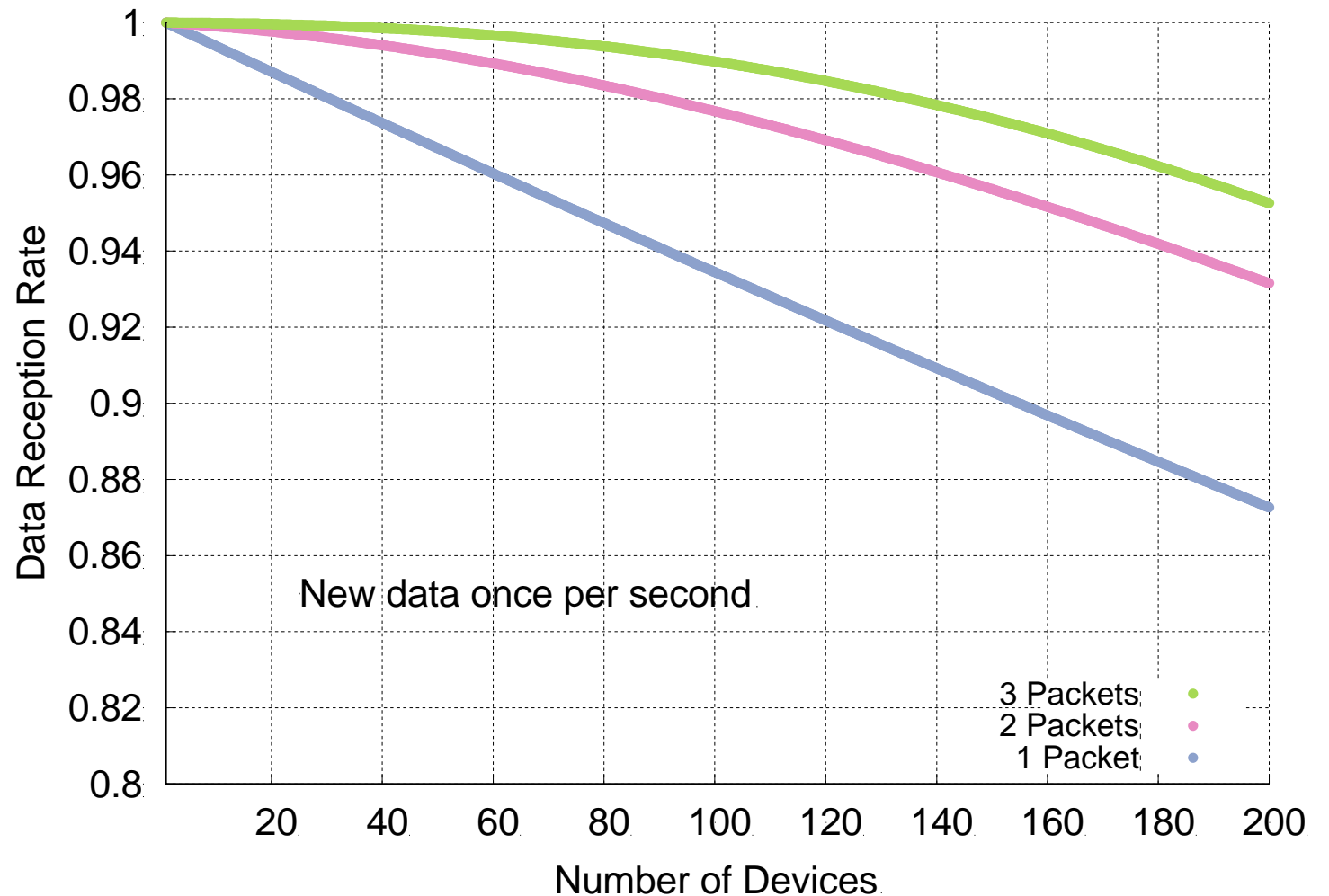
- Probability that at least one of M redundant packets does not have a collision with any of N transmitting devices

$$DRR = 1 - \left(1 - \left(1 - \frac{2 * t_{adv}}{T_{adv_interval} + E[t_{adv_delay}]}\right)^{N-1}\right)^M$$

Redundancy results in high DRR even with many devices.

In this example, a sensor has new data once per second and sends it in 1-3 packets.

Even without redundancy, data reception rates never fall below 87% even with 200 devices in a deployment, assuming no interference.



Redundancy is (normally) better than less congestion.

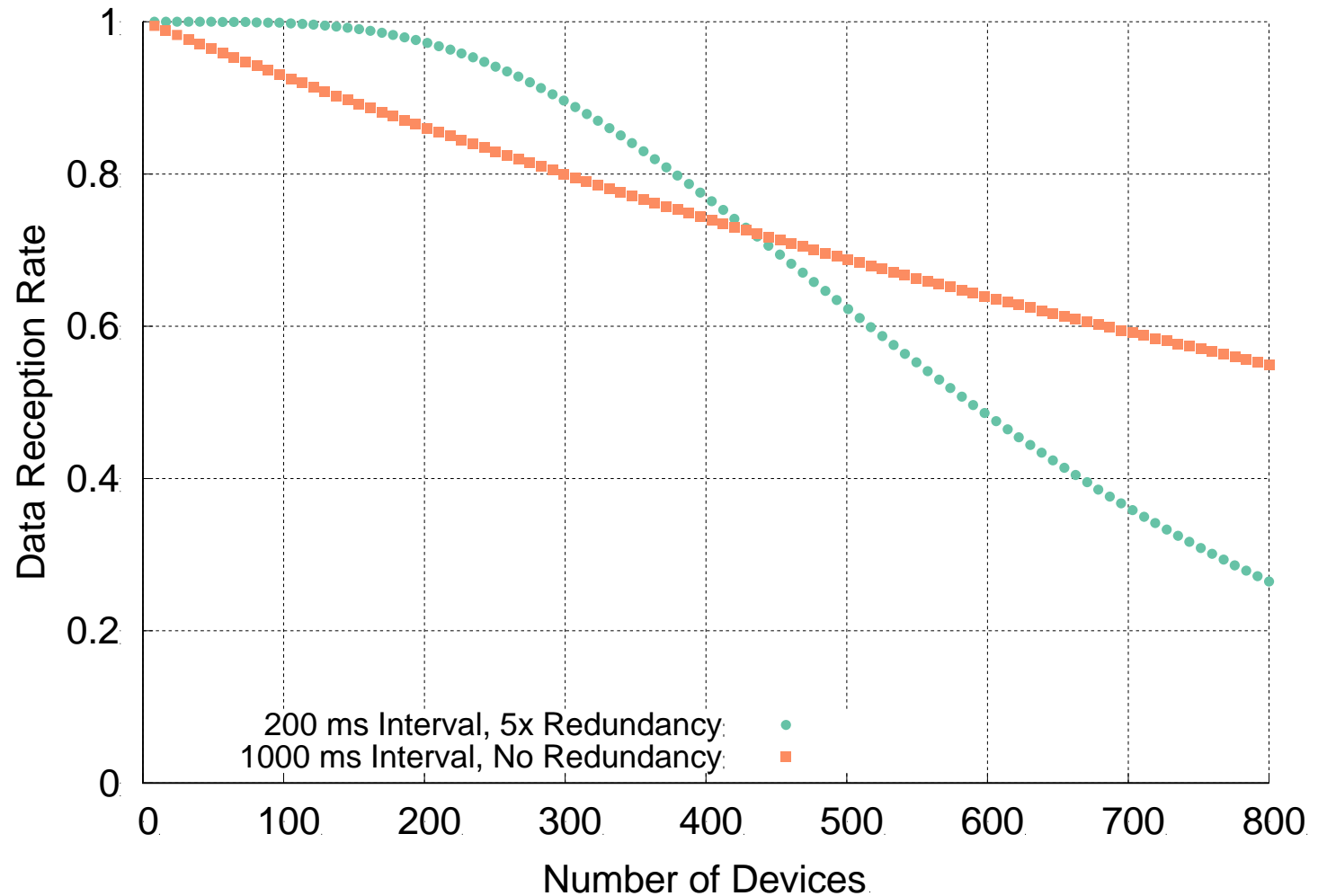
Design question:

- Send more packets to gain from redundancy?

OR

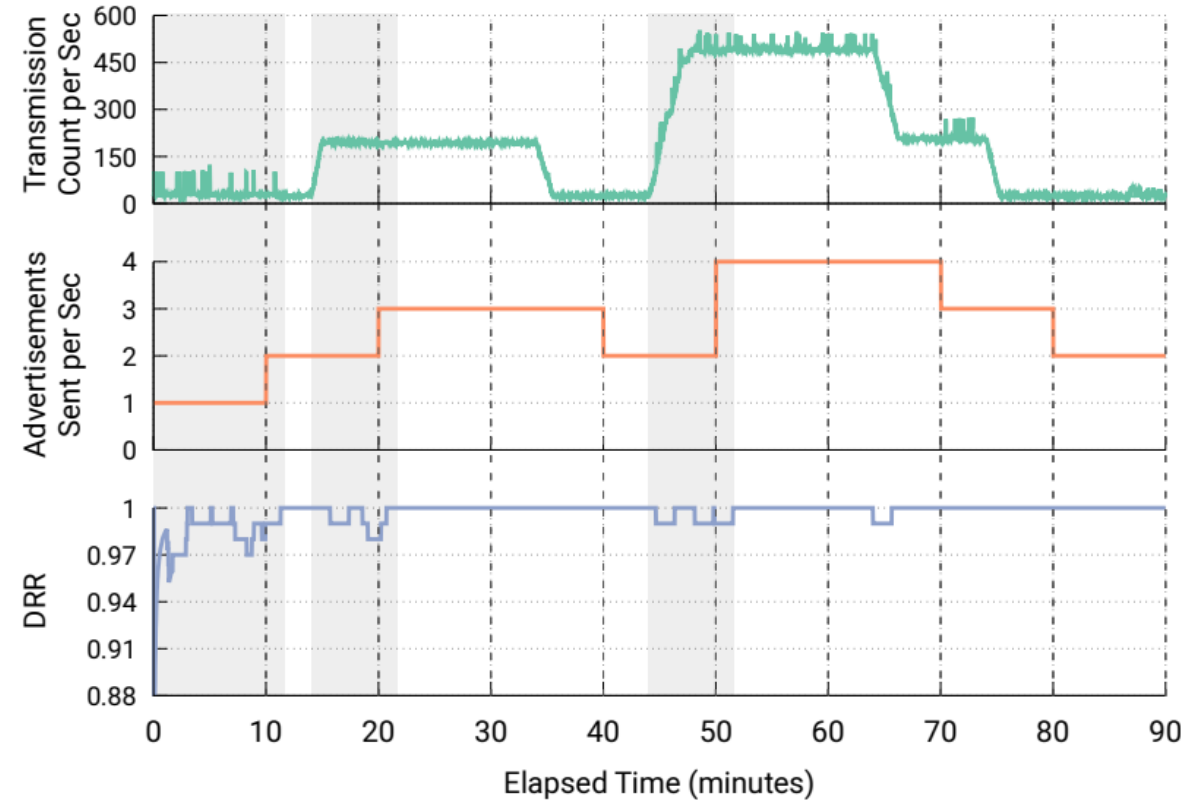
- Send less packets to reduce congestion?

The answer changes,
but only with many
devices.



Automatic adaptation to advertising environment

- Devices could automatically adjust to the environment for best transmission rate
1. Scan and count how many unique devices are seen
 2. Estimate DRR based on that traffic environment for redundancy from 1-10 packets per second
 3. Choose the optimal result



Outline

- BLE Layers
 - Physical Layer
 - Link Layer
- BLE roles
 - Advertising
 - Scanning
 - Scan Responses
- Communicating with advertisements
 - Advertisement Use Cases
 - Energy Use
 - Packet Collisions