CS 439: Wireless Networking

MAC Layer – Bluetooth

 Fall 2024

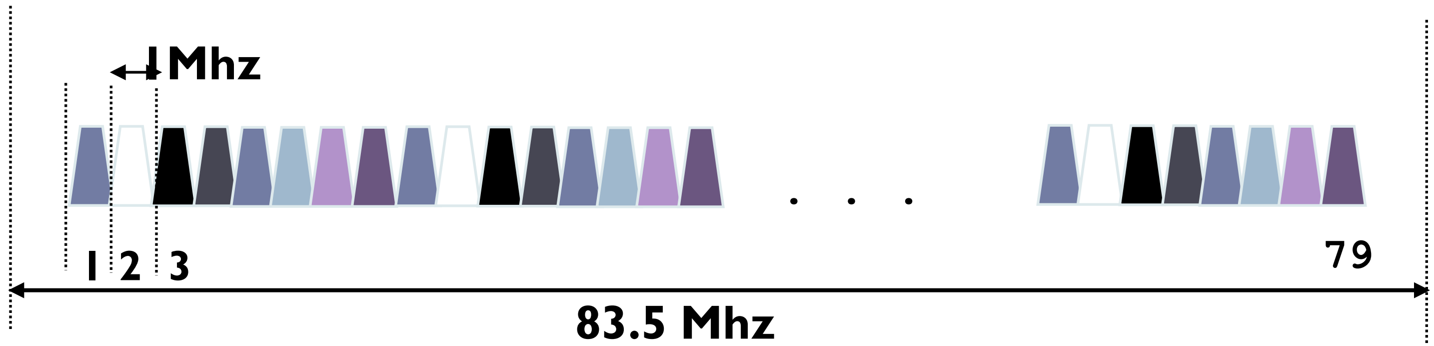
Bluetooth

- Harald Blaatand "Bluetooth" II
  - King of Denmark 940-981 AC
- Runic stones in his capital city of Jelling
  - The stone's inscription ("runes") says:
    - Harald Christianized the Danes
    - Harald controlled the Danes
    - Harald believes that devices shall seamlessly communicate [wirelessly]

Classic Bluetooth

- Cable replacement
  - 2.4 GHz
  - FHSS over 79 channels (of 1MHz each), 1600hops/s
  - 1Mbps
    - Upgraded to 1 or 2 Mbps in 5.0
  - Coexistence of multiple piconets
  - 10 meters (extendible to 100 meters)
    - Max Tx Power 10dB (extendible to 20dB in 5.0)

Bluetooth Radio

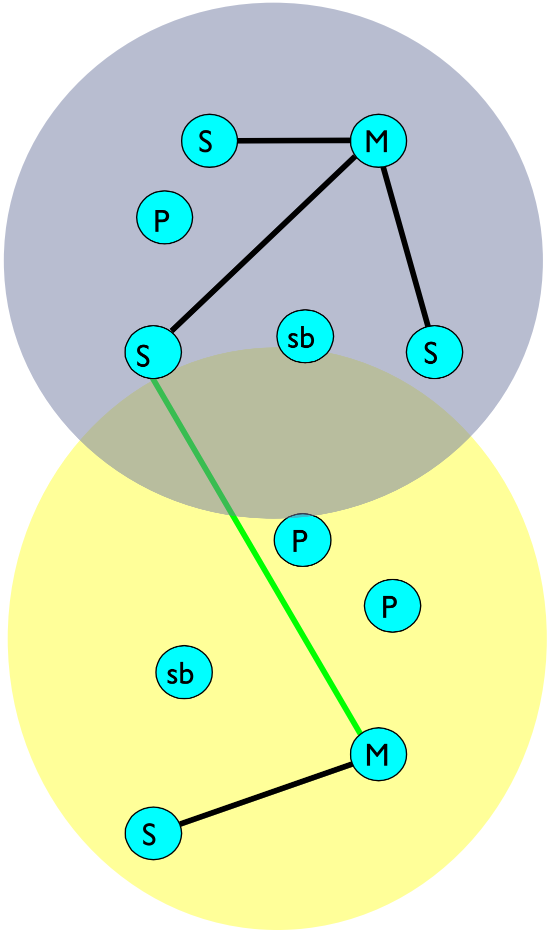


- MA scheme: Frequency hopping spread spectrum.
  - 2.402 GHz + k MHz, k=0, …, 78
  - 1,600 hops per second.
  - 1 Mbps data rate.

  - Upgraded to 2 Mbps in BT 5.0

Bluetooth Network Topology

- **Radio designation**
  - Connected radios can be master or slave
  - Radios are symmetric (same radio can be master or slave)
- **Piconet**
  - Master can connect to 7 simultaneous or 200+ inactive (parked) slaves per piconet
  - Each piconet has maximum capacity (1 Mbps)
  - Unique hopping pattern/ID
- **Scatternet**
  - High capacity system
  - Minimal impact with up to 10 piconets within range
  - Radios can share piconets!



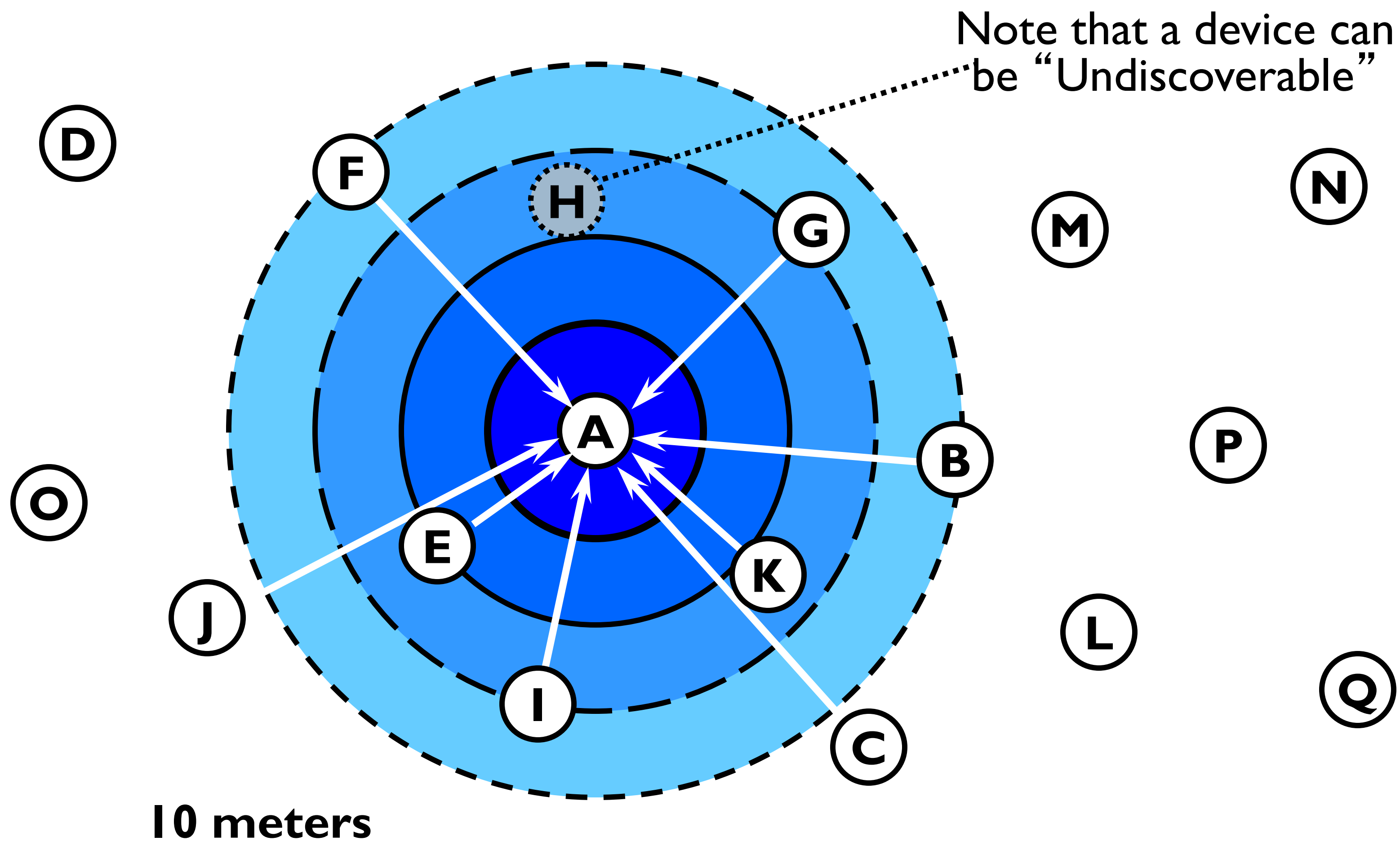
Bluetooth – Contention-free MAC

- Master performs medium access control
  - Schedules traffic through polling.
- Time slots alternate between master and slave transmission
  - Master-slave
    - Master includes slave address.
  - Slave-master
    - Only slave chosen by master in previous master-slave slot allowed to transmit.
  - If master has data to send to a slave, slave polled implicitly; otherwise, explicit poll.

Bluetooth Device Discovery - Inquiry

- Device discovery
  - Sends out an inquire, which is a request for nearby devices (within 10 meters)
  - Devices that allow themselves to be discoverable issue an inquiry response
  - Listeners respond with their address
  - Can take up to 10.24 seconds, after which the inquiring device should know everyone within 10 meters of itself

Bluetooth Device Discovery - Inquiry



After inquiry procedure, A knows about others within range

Bluetooth Inquiry

- Sender
  - Inquiry sent on 16 different frequencies
  - 16 channel train
    - about 1.28 seconds per channel
    - One full 16 channel train takes 10ms
- Receiver (device in standby mode)
  - Scans long enough for an inquiring device to send the inquiry on 16 frequencies
  - Scan must be frequent enough to guaranteed wake up during a 16 channel train
    - Enters inquiry scan state at least once every 1.28 seconds, and stays in that state for 10ms

Bluetooth Inquiry - Reliability

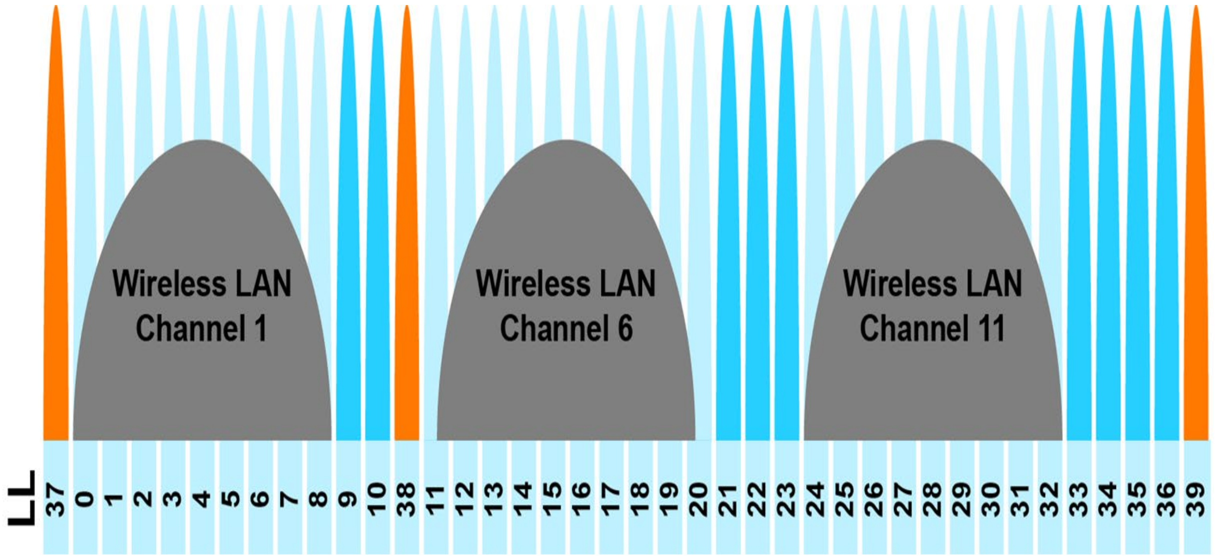
- Challenge
  - Noisy channels
  - Lost packets
    - Train scan is repeated up to 4 times for each train (10.24 seconds)
    - Designed to successfully communicate at least once with all devices within range

BLE Highlights

- Shared wireless channel
  - BLE operates in the 2.4 GHz ISM band with Wi-Fi and other technologies (phones, microwave ovens …)
- BLE = Bluetooth Low Energy
  - Improved discovery
  - Key component: Beacons
    - Tags send out advertising beacons (typ. dist 30ft)
    - Phones scan for beaco

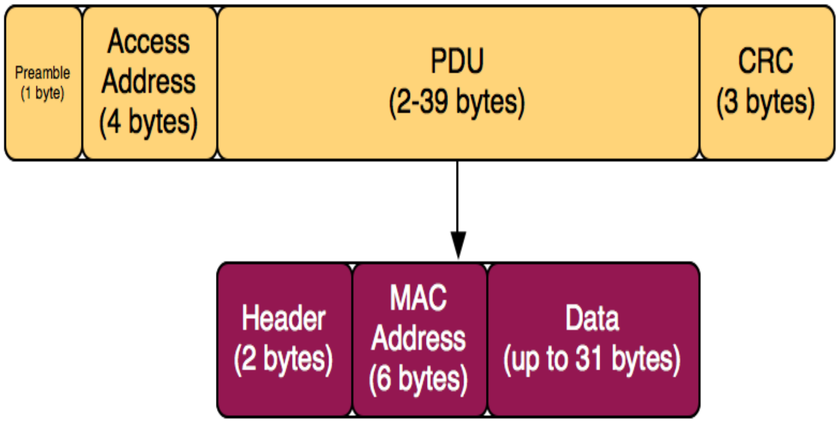
BLE Highlights: Channel Use and Coexistence with Wi-Fi

- Separate advertising and connected channels
  - Key: Three disjoint advertising channels (37, 38, 39)
  - Positioned between Wi-Fi channels (1, 6, 11)



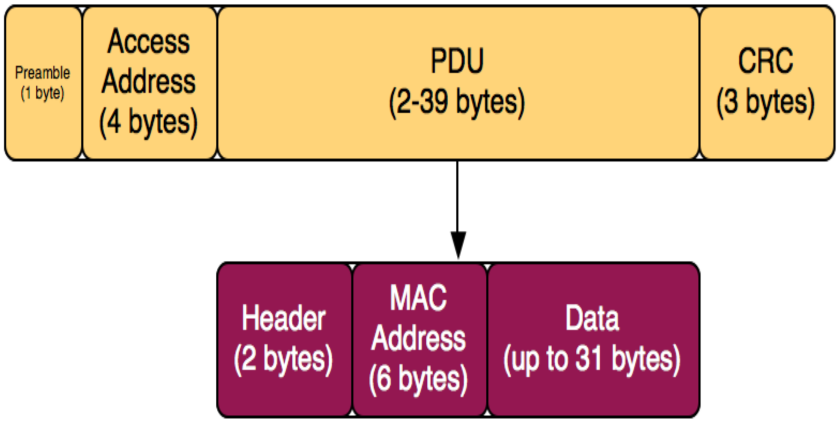
BLE Highlights: Advertising

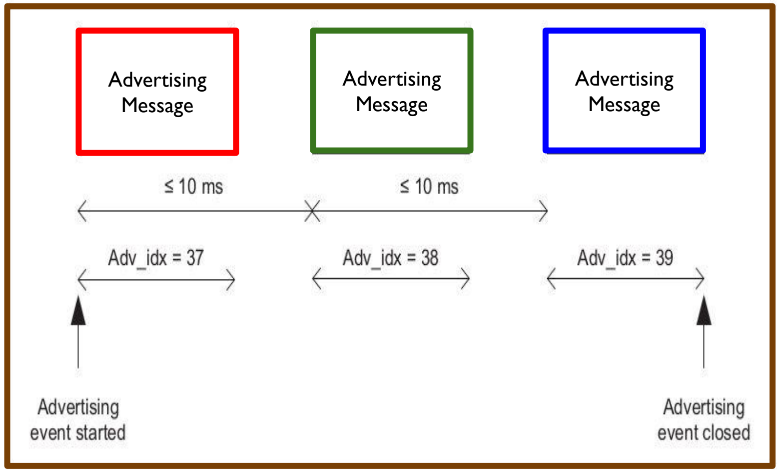
- Advertising Tags
- Advertising Messages
  - Header + MAC Address + up to 31 Bytes of data
    - ~200 - 400 usec per packet
  - Two types: Non-scannable, Scannable



BLE Highlights: Advertising

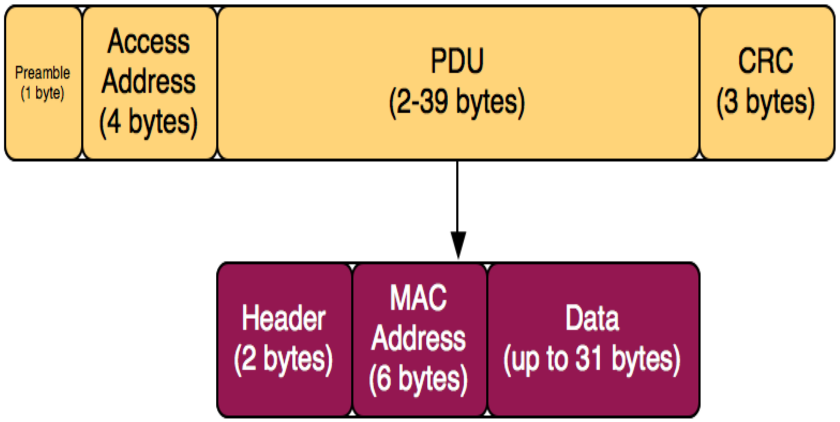
- Advertising Tags
- Advertising Messages
  - Header + MAC Address + up to 31 Bytes of data
    - ~200 - 400 usec per packet
  - Two types: Non-scannable, Scannable
- Advertising Event
  - One advertising message sent out on each advertising channel (37, 38, 39)

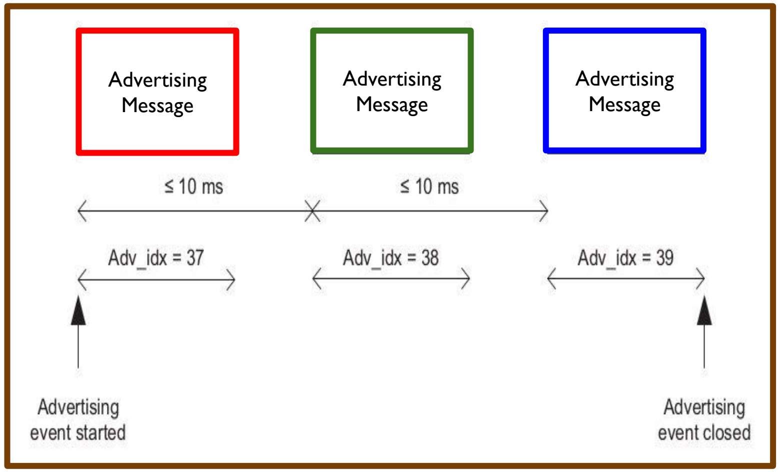




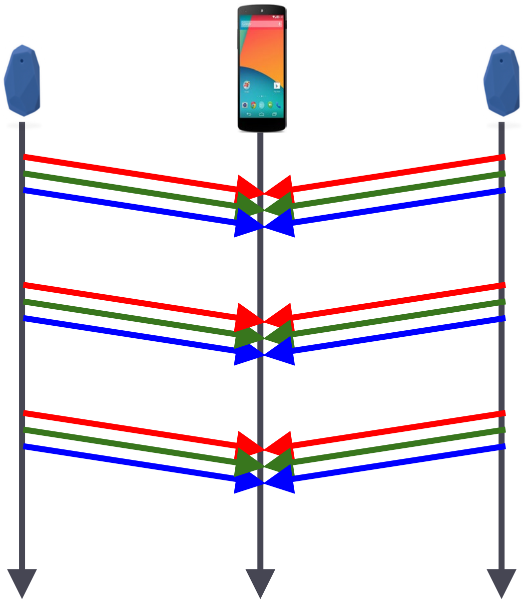
BLE Highlights: Advertising

- Advertising Tags
- Advertising Messages
  - Header + MAC Address + up to 31 Bytes of data
    - ~200 - 400 usec per packet
  - Two types: Non-scannable, Scannable
- Advertising Event
  - One advertising message sent out on each advertising channel (37, 38, 39)
- Advertising Interval
  - One advertising event per advertising interval
  - e.g., every 1 sec or 100 msec



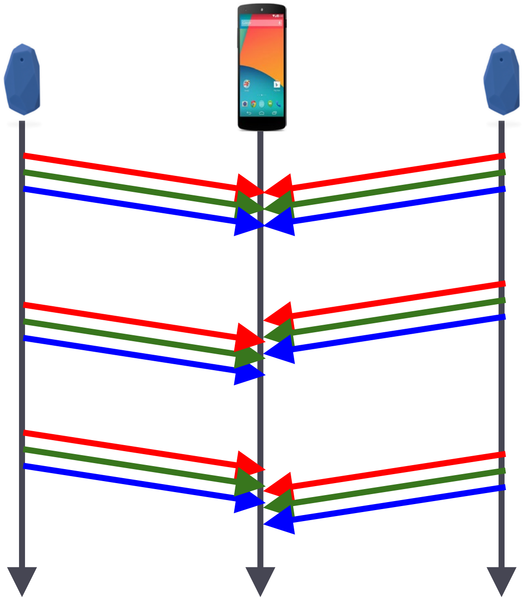


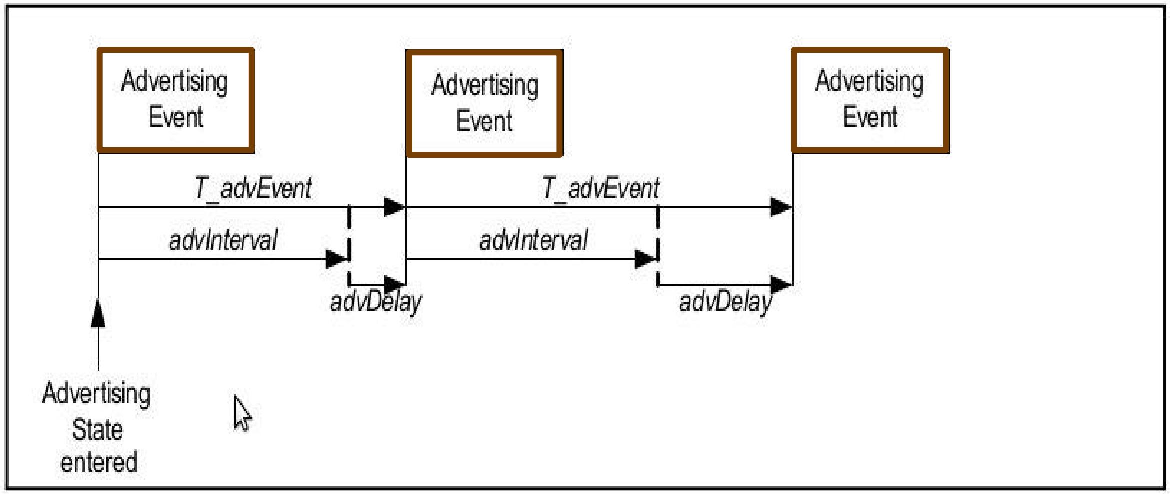
BLE Highlights: Advertising and Collisions



- If tags get synchronized, all advertising messages will collide

BLE Highlights: Advertising and Collisions

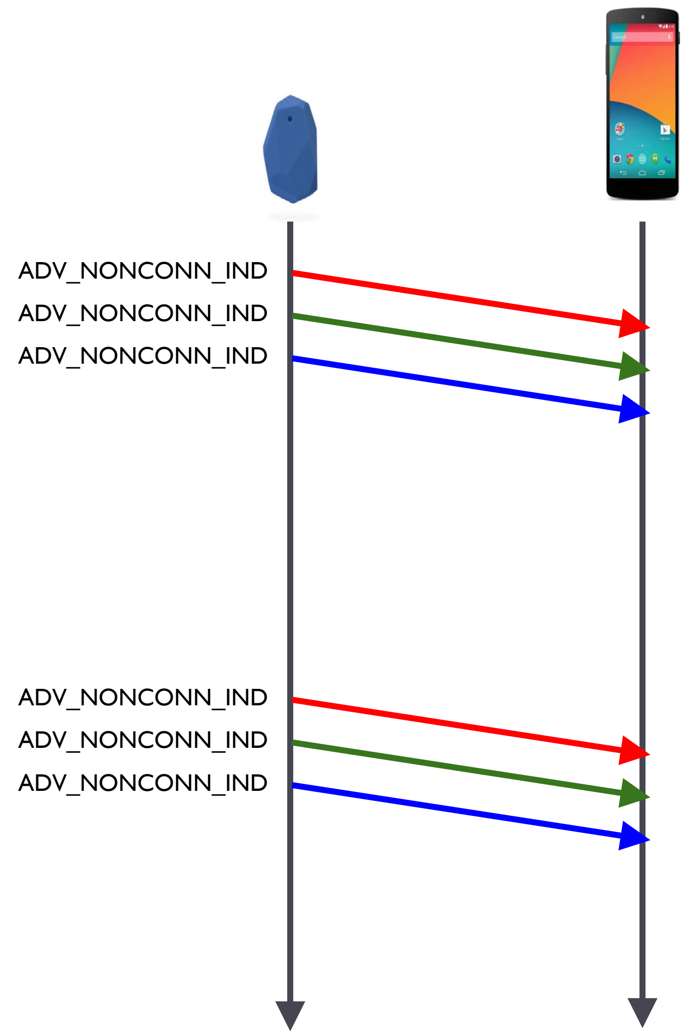


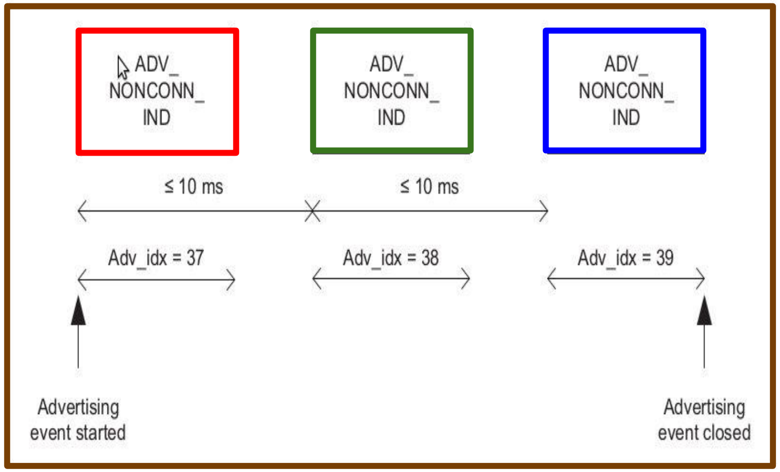


- Collision avoidance
  - Jitter advertising times
  - advDelay is added on to the end of each advertising event
  - advDelay = rand [0,10ms]

BLE Highlights: Tags Types - Non-Scannable

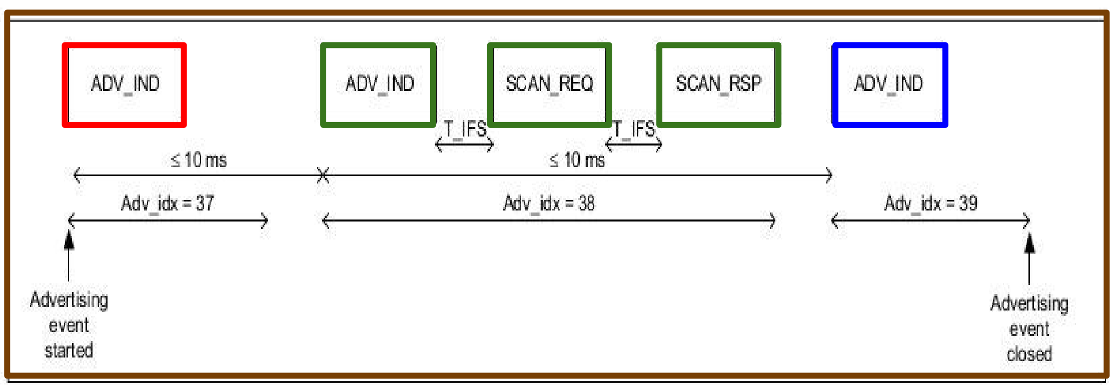
- Non-Scannable Tags
- Ex. gBeacon v3, iBeacon (?)
- Tags send ADV_NONCONN_IND messages
- Typically sent back-to-back
- Scanners listen, but do not respond

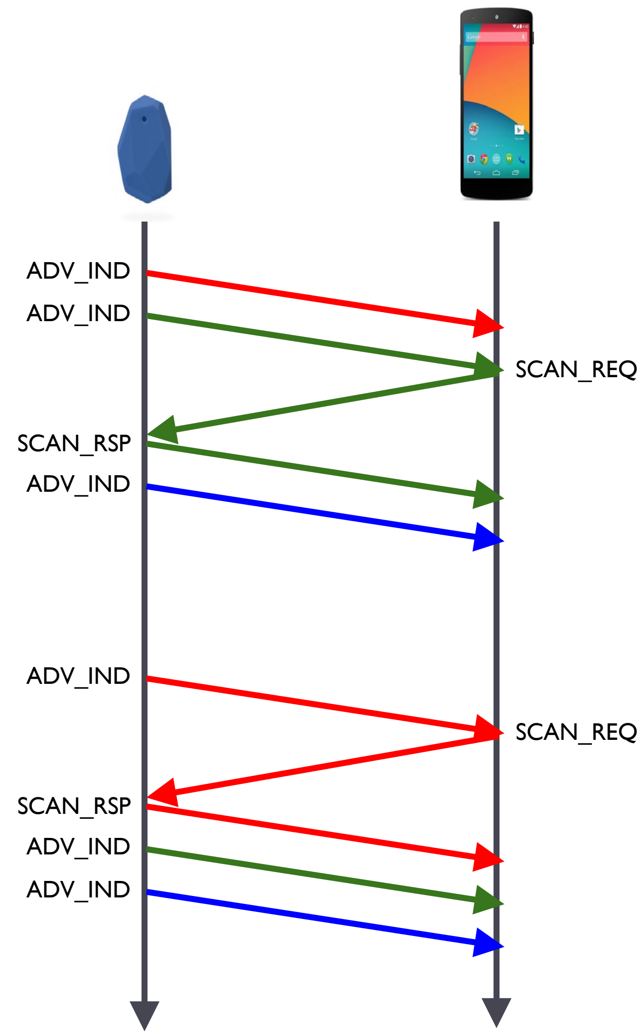




BLE Highlights: Tags Types - Scannable

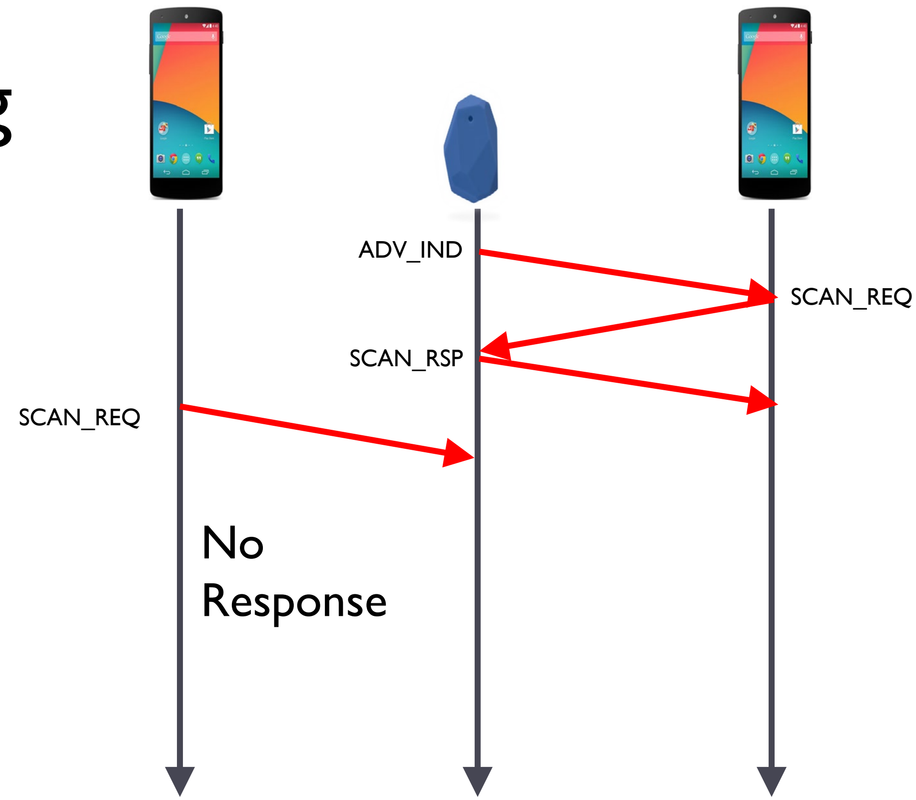
- Scannable Tags
- Ex. gBeacon V1, Estimote
- Tags send ADV_IND messages
- Scanners respond with SCAN_REQ message
- Tags respond with SCAN_RSP message
  - Up to 31 Bytes of extra data
- Tags wait ~150 usec for a request after beacon





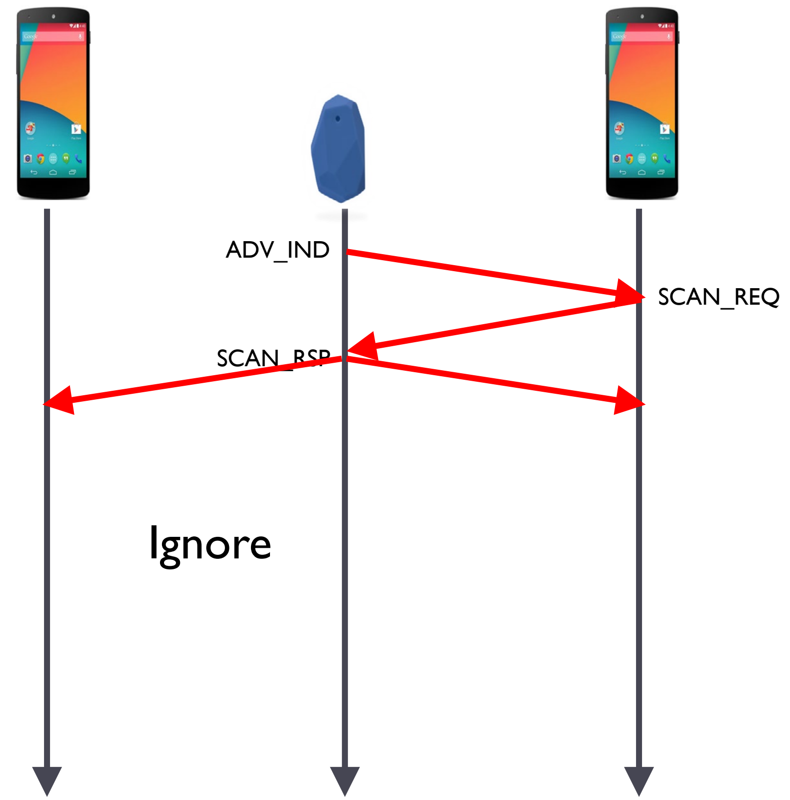
Scannable Tags

- One SCAN_RSP per channel per advertising event



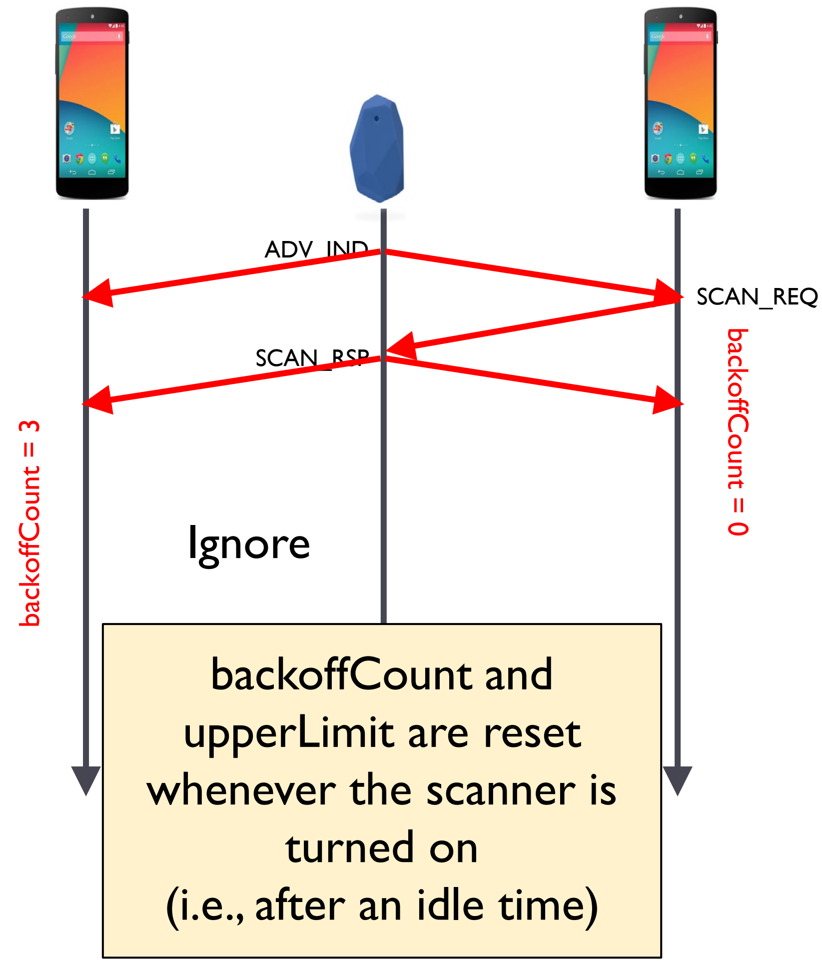
Scannable Tags

- ONLY accept SCAN_RSP if SCAN_REQ was sent to that tag on that channel during that advertising event
- Some collision tolerance
  - Any requesting scanner can receive a SCAN_RSP as long as one SCAN_REQ is received and the tag responds
  - BUT, No SCAN_RSP if all SCAN_REQs collide



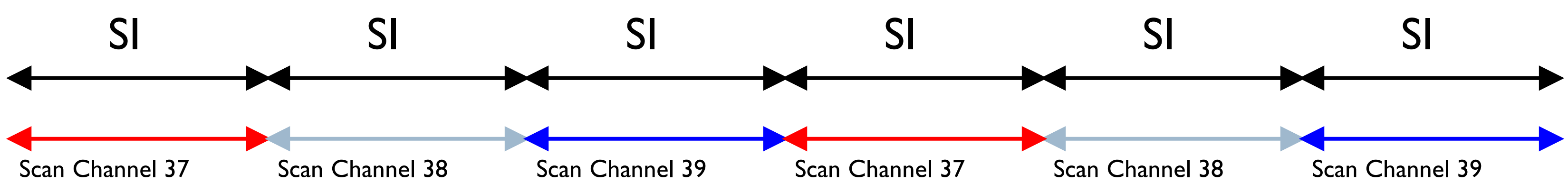
SCAN_REQ Collision Avoidance

- **Scanner backoff procedure**
  - Two parameters
    - backoffCount, upperLimit
  - On starting scan
    - upperLimit = 1, backoffCount = 1
  - Decrement backoffCount on receipt of ADV message
    - Only send SCAN_REQ if backoffCount == 0
  - Adapt upperLimit based on success or failure of receipt of SCAN_RSP
    - Reset backoffCount
    - backoffCount = rand (1, upperLimit)



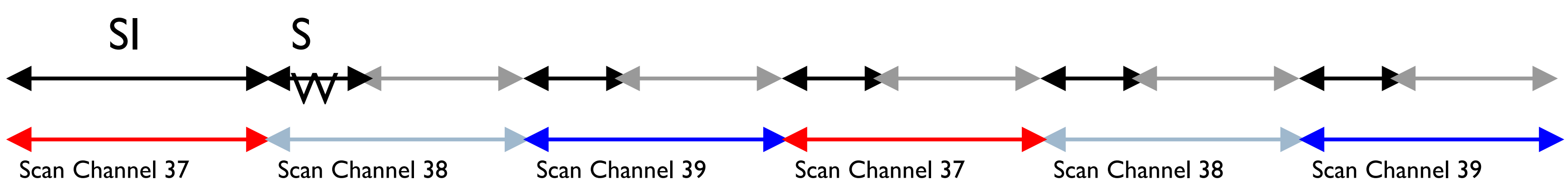
BLE Highlights: Low-level Scanning

- Scanners
- Scan for tags on sequential channels (37, 38, 39)
- Scan Interval (SI)
  - Time spent on a channel



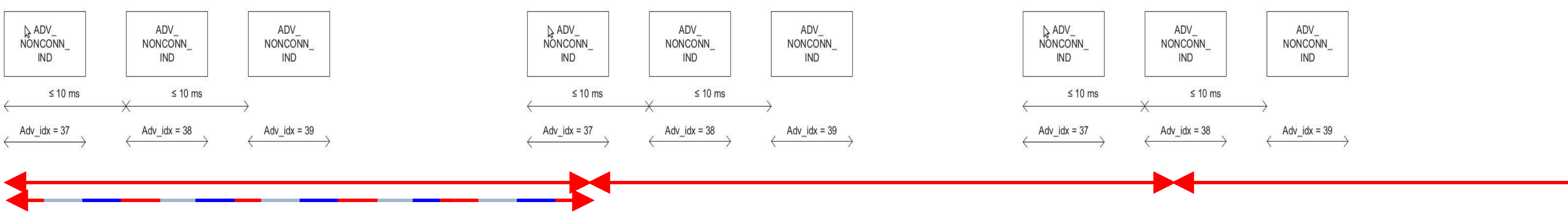
BLE Highlights: Low-level Scanning

- Scan Time
  - Scan Int == Scan Window ⇒ Always on
- Scanners
- Scan for tags on sequential channels (37, 38, 39)
- Scan Interval (SI)
  - Time spent on a channel
- Scan Window (SW)
  - Time spent scanning at beginning of Scan Interval



BLE Highlights: Application-level Scanning

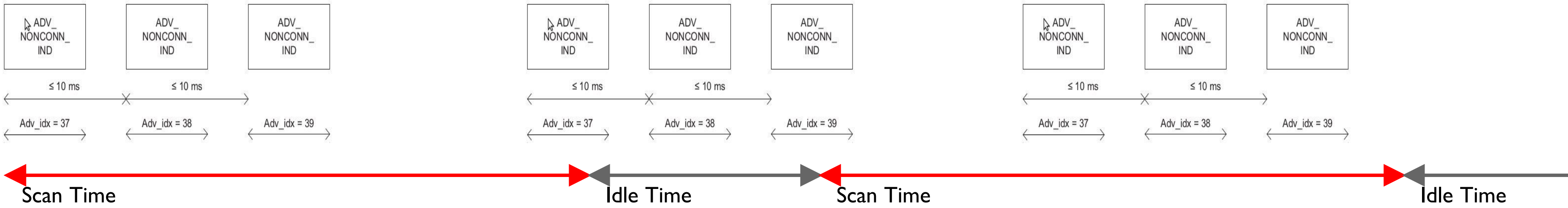
- Scanners
- Application Scan Time
  - > Tag Advertising Interval



Application Scan Time

BLE Highlights: Application-level Scanning

- Scan Time
  - 100% on Idle Time = 0
- (Continuous scanning)
  - 10% on Idle Time = 10 * Scan Time
- Scanners
- Application Scan Time
  - > Tag Advertising Interval
- Application Idle Time



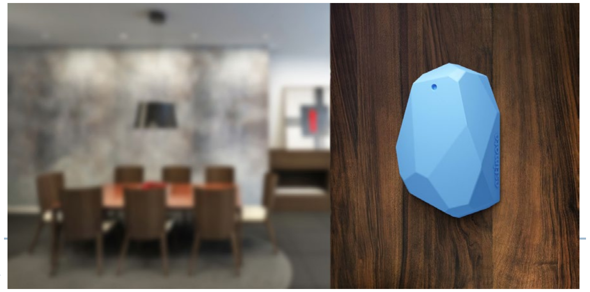
BLE Highlights: MAC Behavior

- No Carrier Sense
  - Tag does not listen for a clear channel before sending any message
- Minimal Contention Avoidance
  - Jitter length of advertising interval + rand [0, 10 ms]
  - Backoff for sending SCAN_REQ

- Other parameters
  - Inter-frame spacing 150us (from spec)
  - Channel switching delay 274us (from Nordic)
  - Scan Interval 11.25ms (from spec/Nexus 5)
  - Scan Window 11.25ms (continuous scanning)

BLE in the Real World

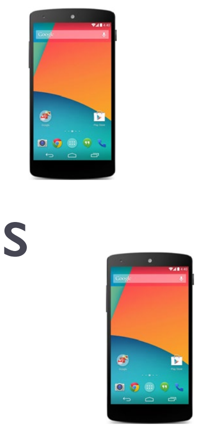
- BLE beacons (or tags)
  - Location-specific information
  - Deployed in public places
    - Stores, airports, museums
  - Accessed via phones with BLE

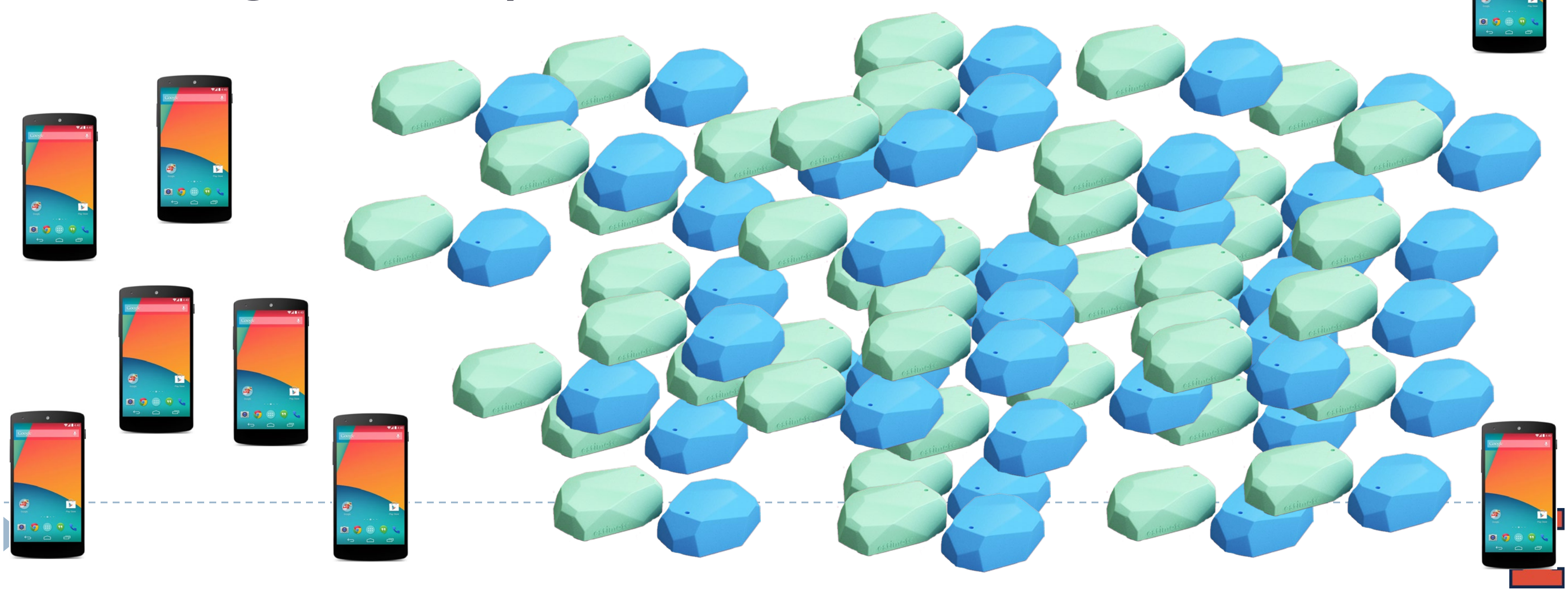


- Performance questions
  - How long does it take to detect a nearby tag?
  - Can we detect a tag within 5 sec with 95% success?

BLE in the Real World - Density

- As deployments increase, how will the tags behave?
  - What is the effect of high density tags and phones on tag discovery?



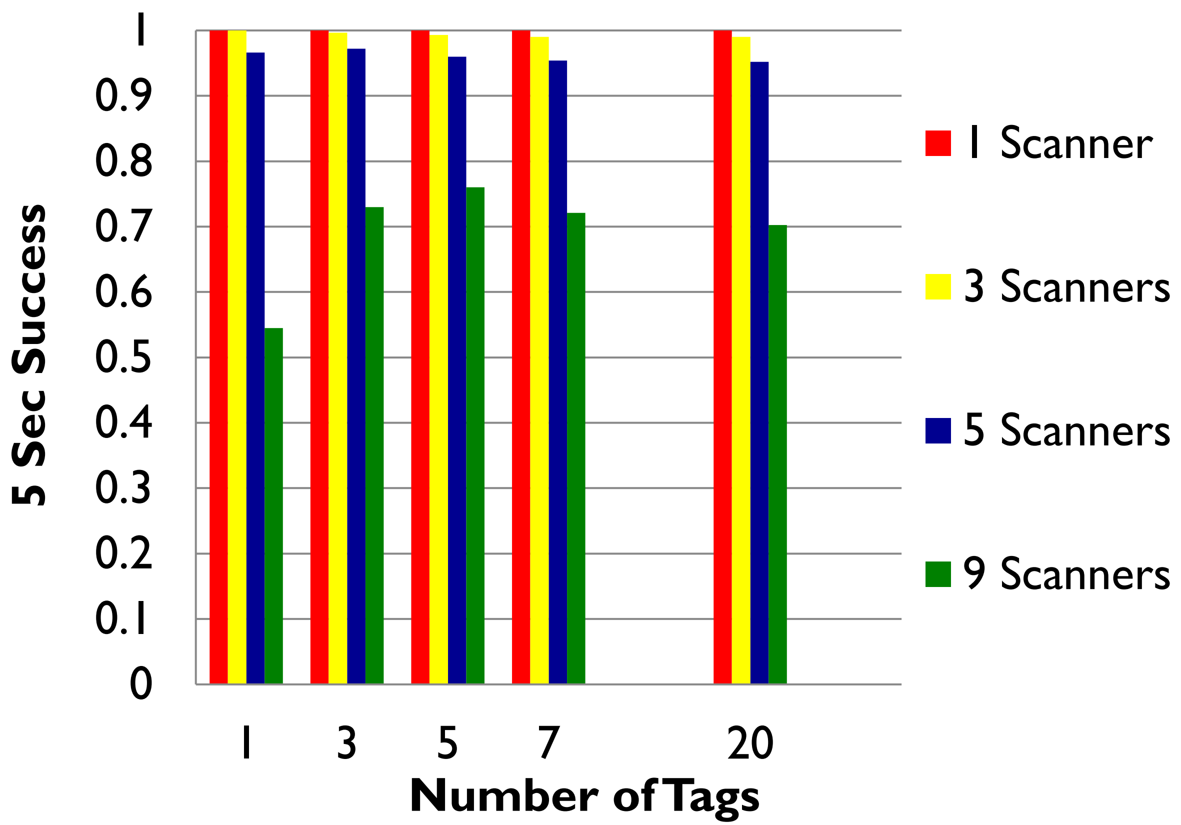


Evaluating Tag Behavior

- Environmental Impact
  - At what density of tags or phones does the system break down?
- Metric
  - 5 Sec Success
    - Could the tag be found in 5 sec?
    - Checked every 1 sec over the whole run

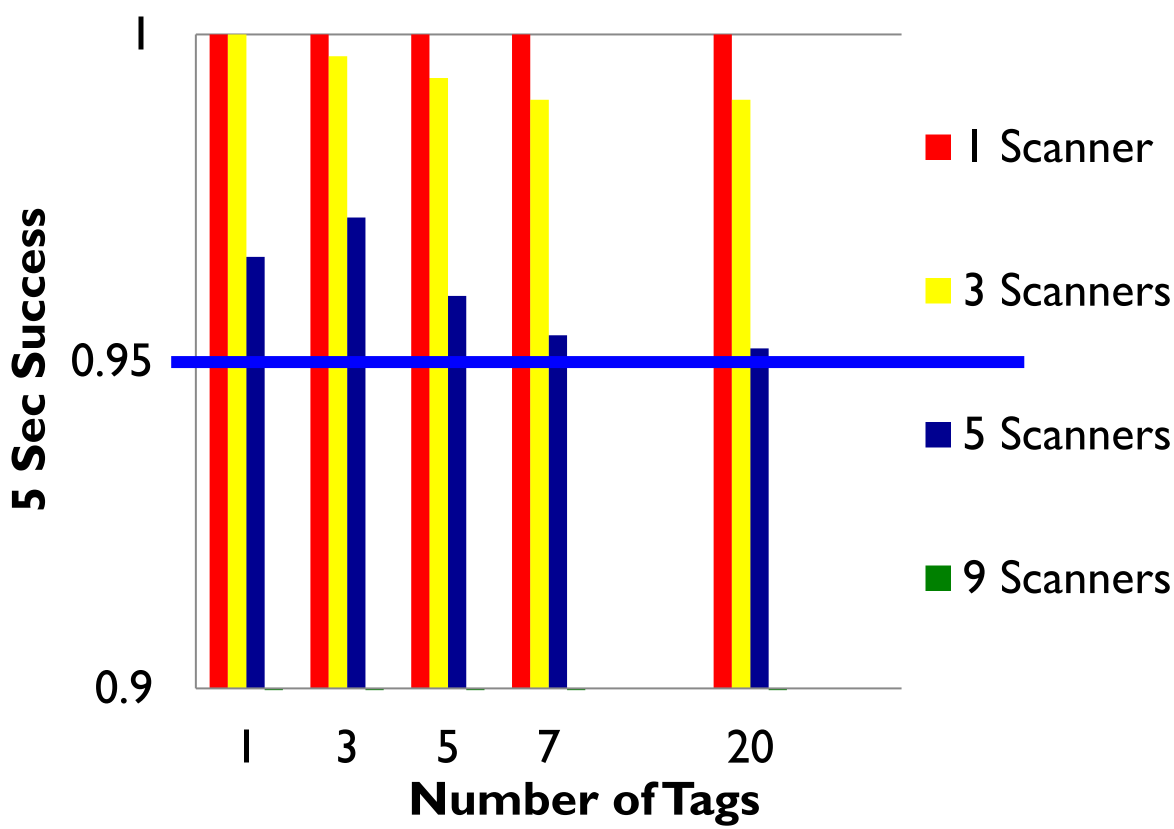
Evaluation: BLE Scan/Response

- 5 second success
  - Multiple chances to find the tag
  - Success decreases significantly as more phones are added
  - Number of phones is more important than number of tags



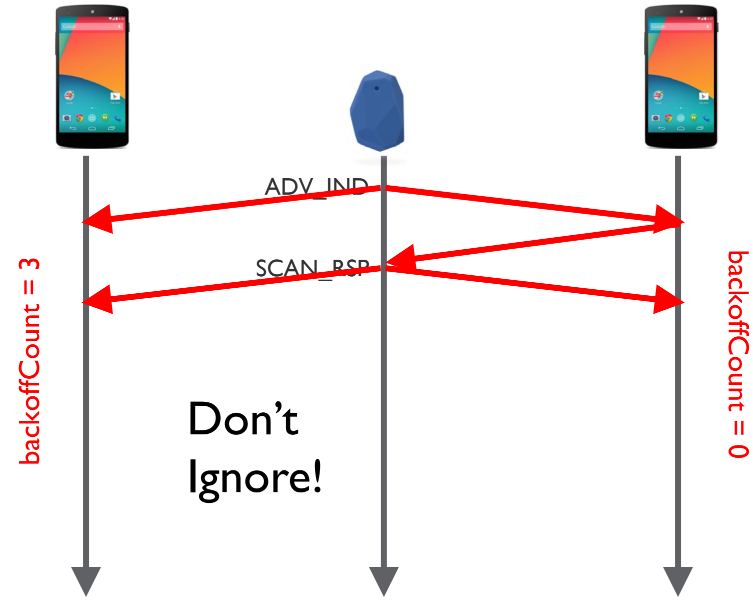
Evaluation: BLE Scan/Response

- 5 second success
  - Below target threshold for more than 5 phones



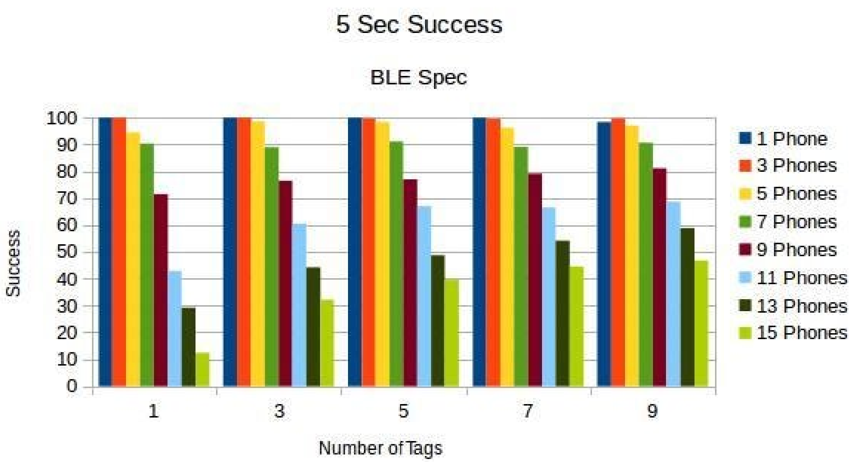
SCAN_REQ: Opportunistic Listening

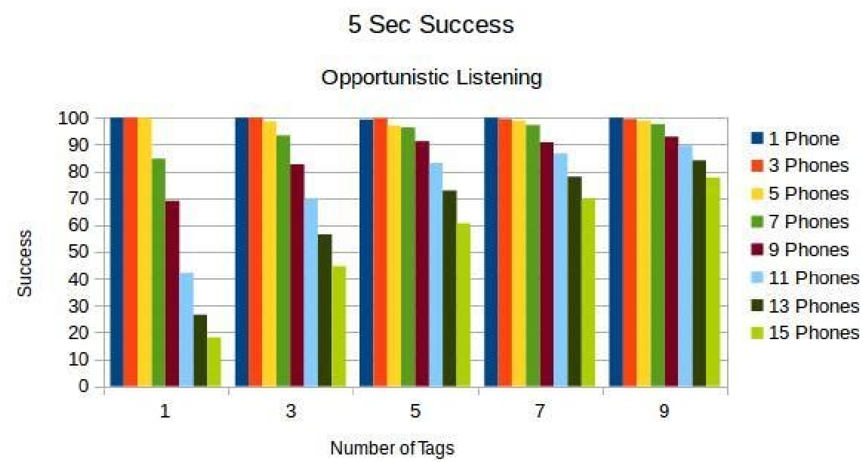
- Accept a SCAN_RSP on a channel if a SCAN_REQ would have been sent, but the backoff procedure indicated not to send it
  - Any requesting or backing off scanner can receive a SCAN_RSP as long as one SCAN_REQ is received and the tag responds
  - Still, No SCAN_RSP if all SCAN_REQs collide



Opportunistic Listening: Simulation Comparison

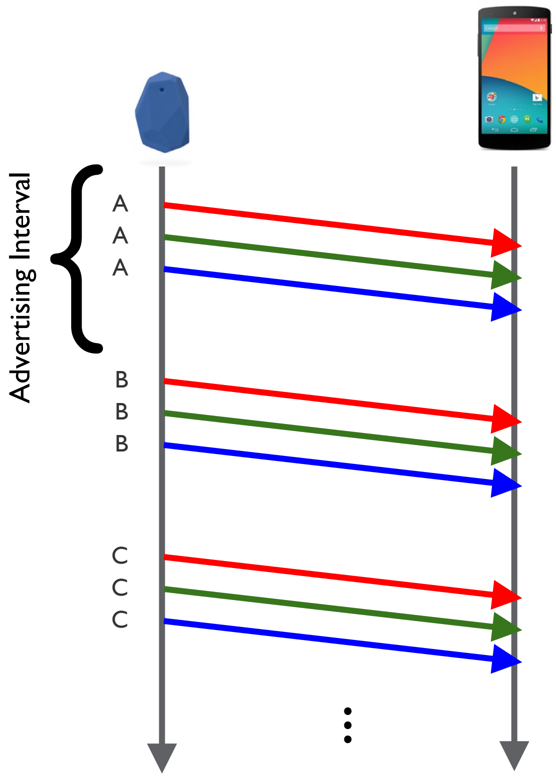
- Significant increase in success rate as number of phones increases
- Cannot prevent SCAN_REQ collisions





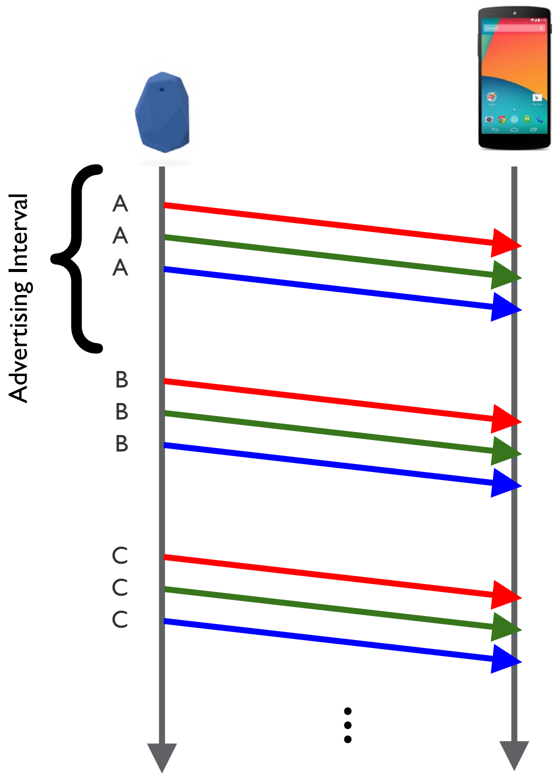
Beacon Trains

- Approach
  - Advertising data is broken in n beacons
    - For n = 5: A,B,C,D,E
    - Max 31 bytes
  - Alternate sending data in non-scannable beacons
    - Send A in 1st advertising interval, Send B in 2nd advertising interval, etc
    - Must receive all n to get complete advertising data



Beacon Trains

- Approach
  - Pro
    - Advertising data = n * 31
  - Con
    - Delay to receive all n packets in a train

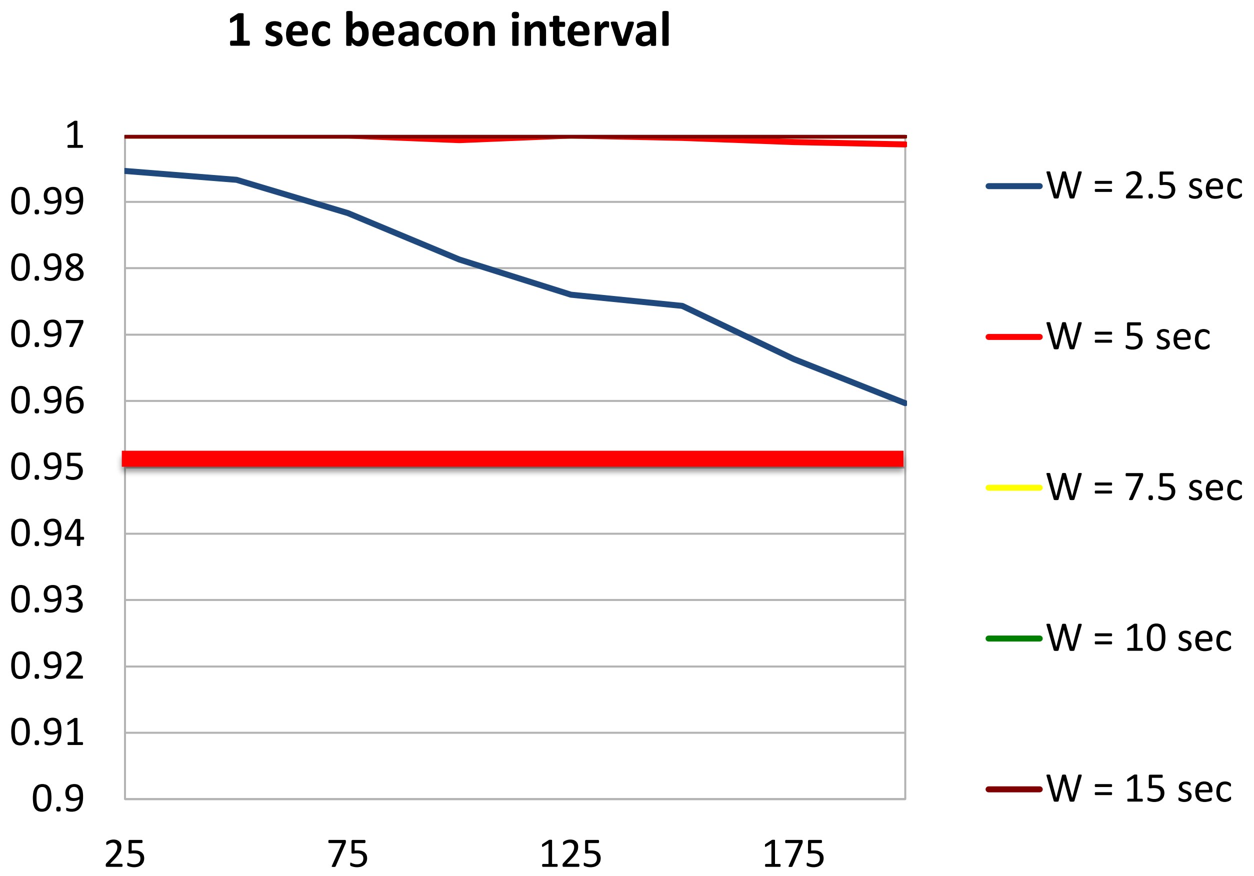


Evaluation

- Goal
  - Evaluate effect of higher tag density on tag discovery for beacon trains
- Parameters
  - Number of tags: 25-200
  - Beacon interval: 1s, 500ms, 250ms, 100ms
- Metric
  - t Sec Success: are all packets from the train received in t sec?
- Discrete Event Simulator
  - ns-3: No true BLE
  - Based on Zigbee PHY layer adapted to 1Mbps channel

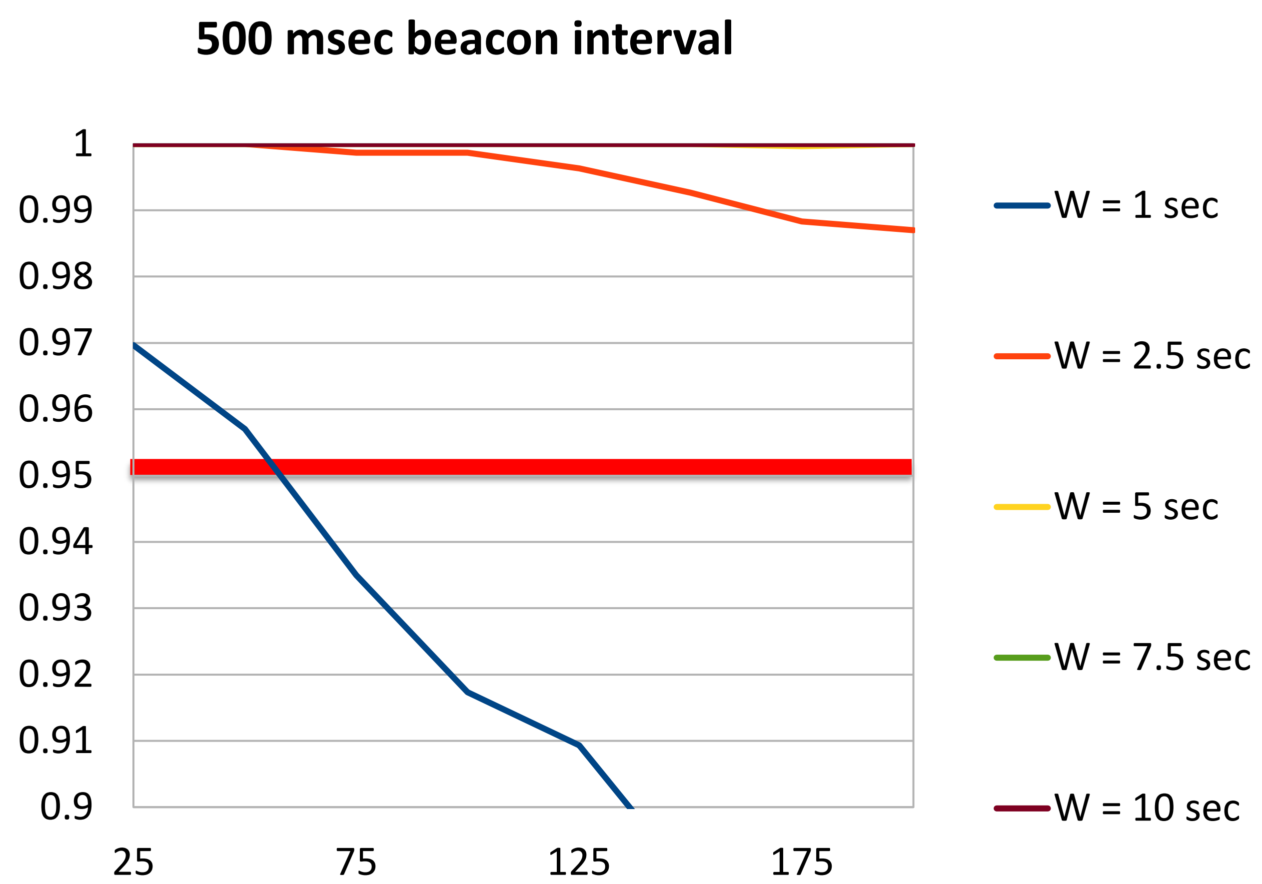
Passive Beacons: Success

- Density has little effect above a 5 sec wait time
- Success is still over 95% with a 2.5 sec wait time
- Wait ~2.5x AdvInt



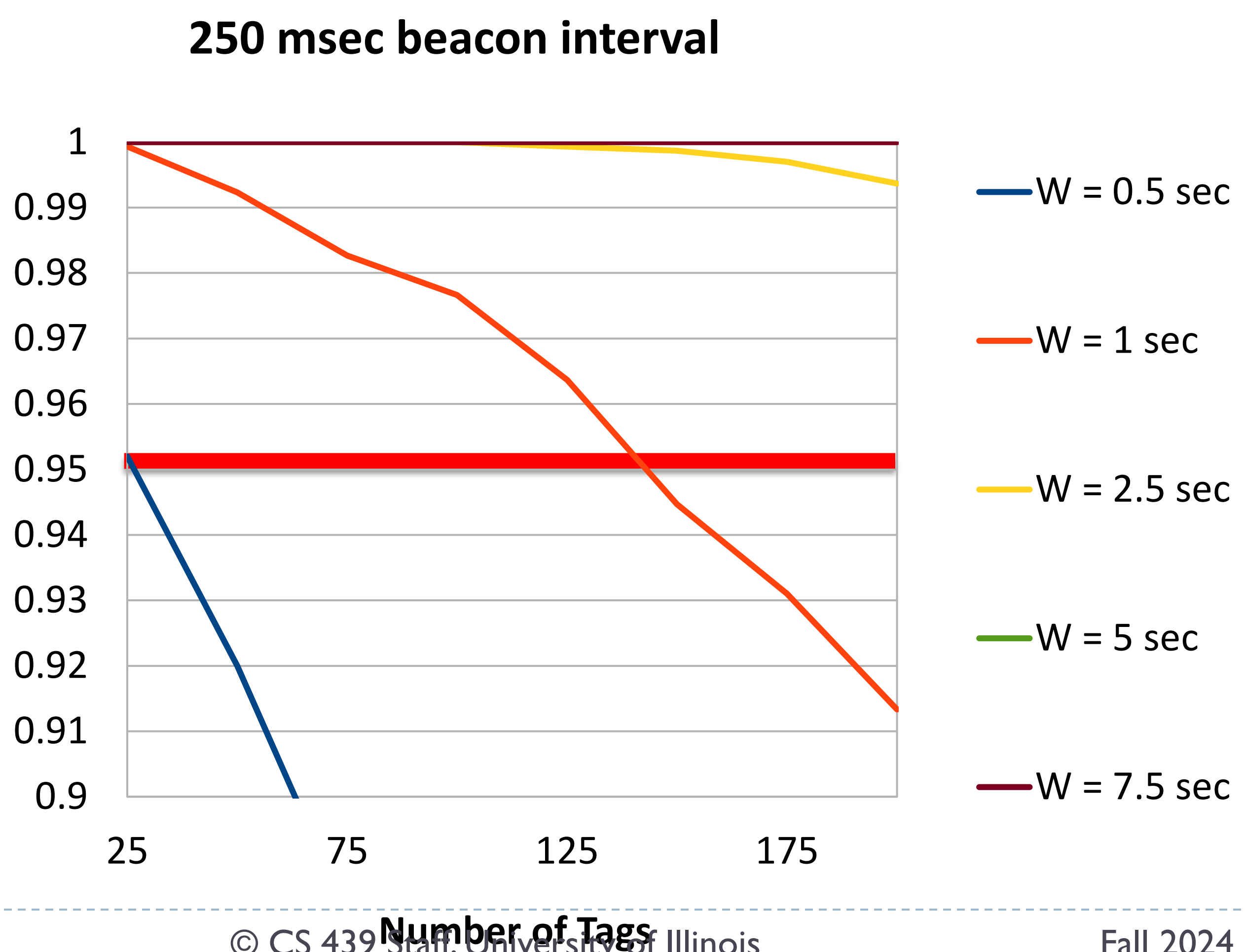
Passive Beacons: Success

- Density has little above a 2.5 sec wait time
- Success quickly drops below 95% with a 1 sec wait time
- Wait ~5x AdvInt



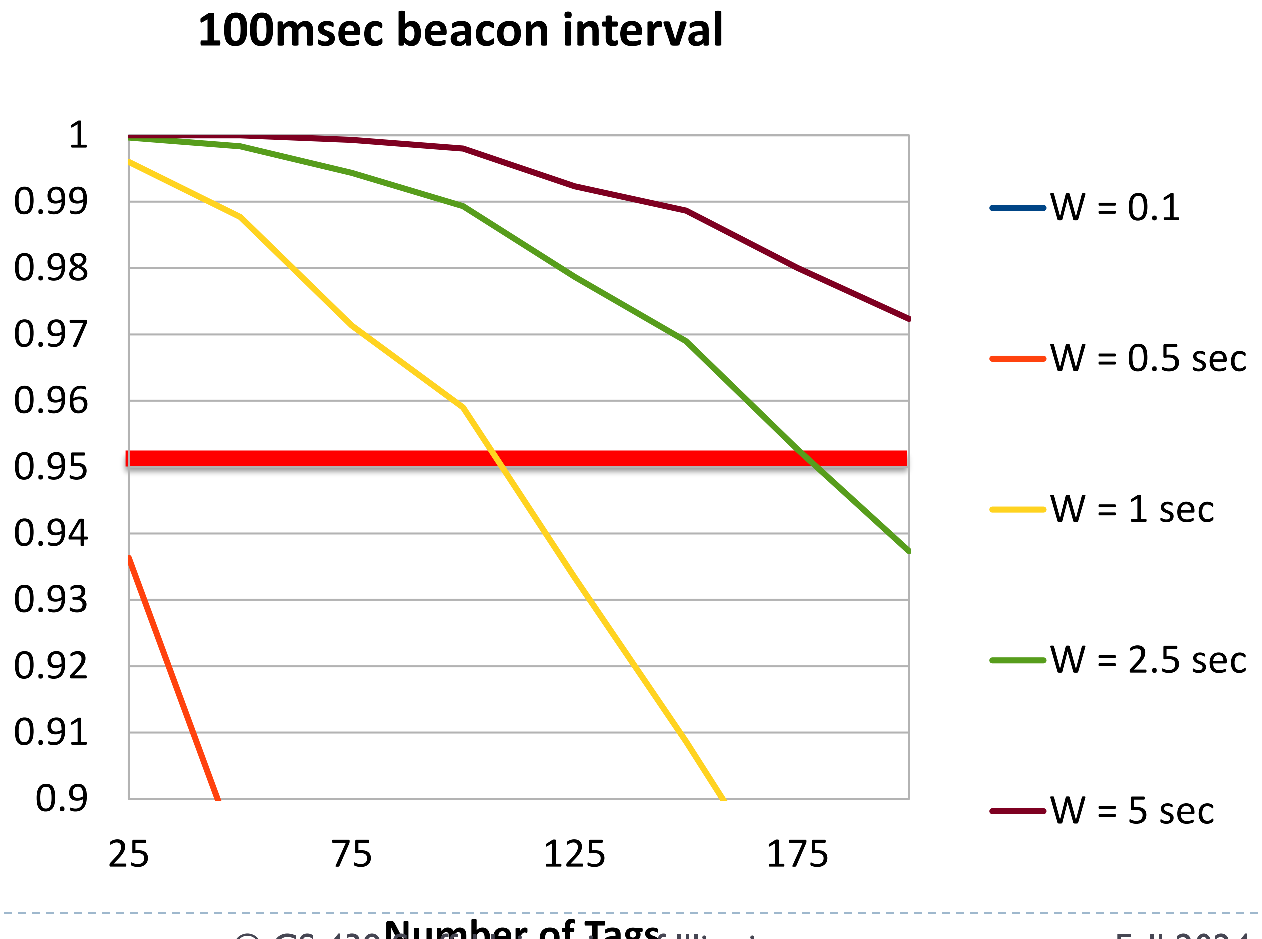
Passive Beacons: Success

- Density has more effect, reducing the success of 1 sec wait time
- Success is almost always below 95% with a 0.5 sec wait time
- Wait ~4x AdvInt



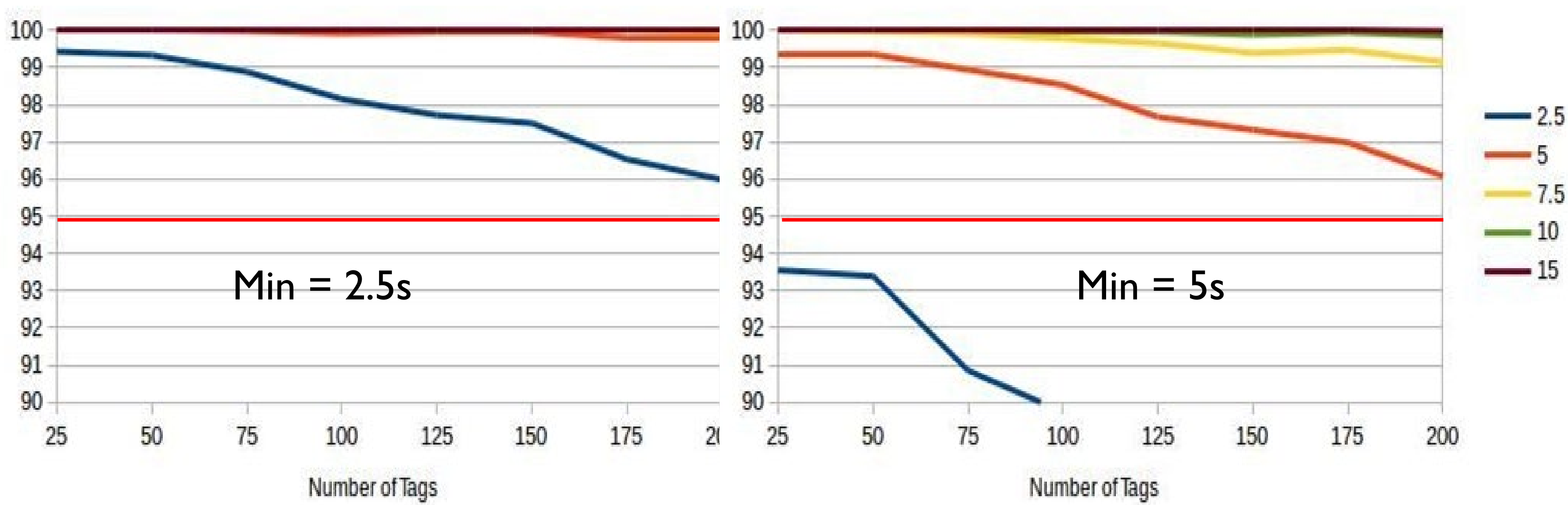
Passive Beacons: Success

- Density has the worst effect
- Only 5 sec wait time has effective success
- Even 2.5 sec wait time drops below 95%
- Wait > 10x AdvInt



Beacon Train Success - 1 sec BI

N = 1 N = 2

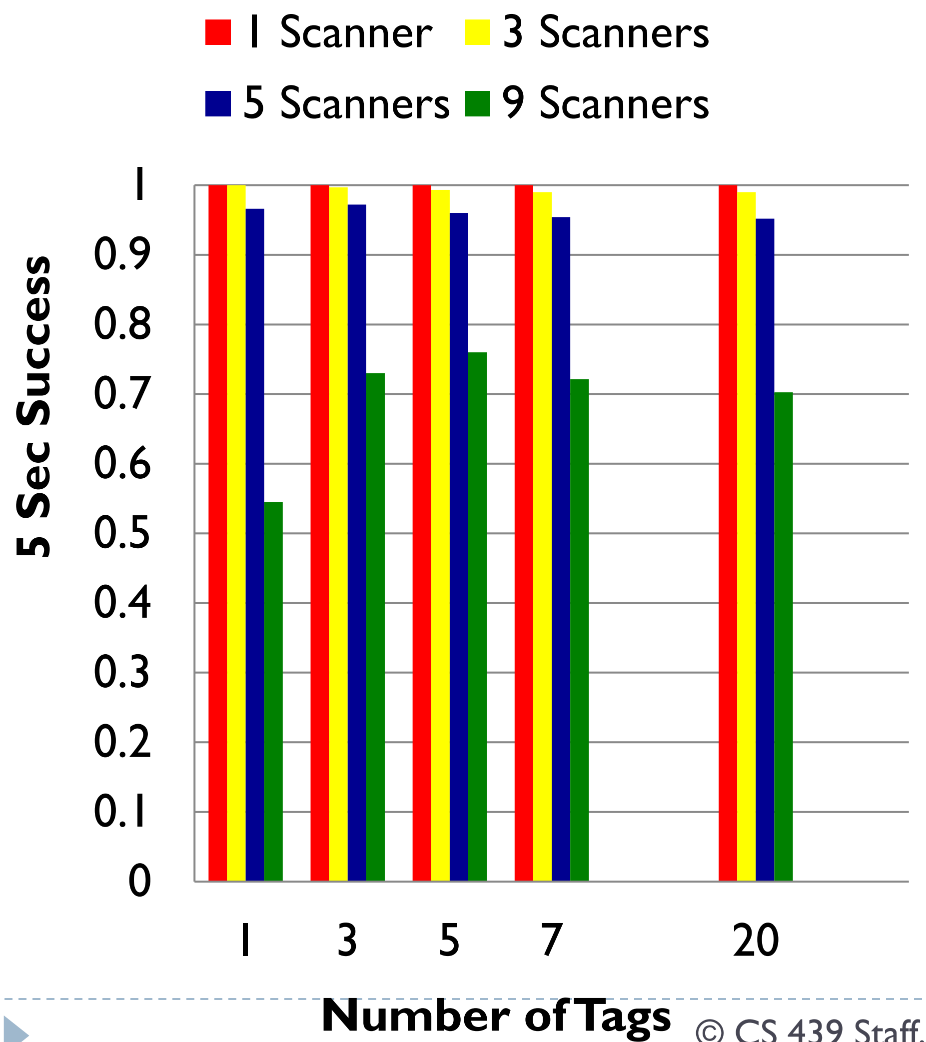


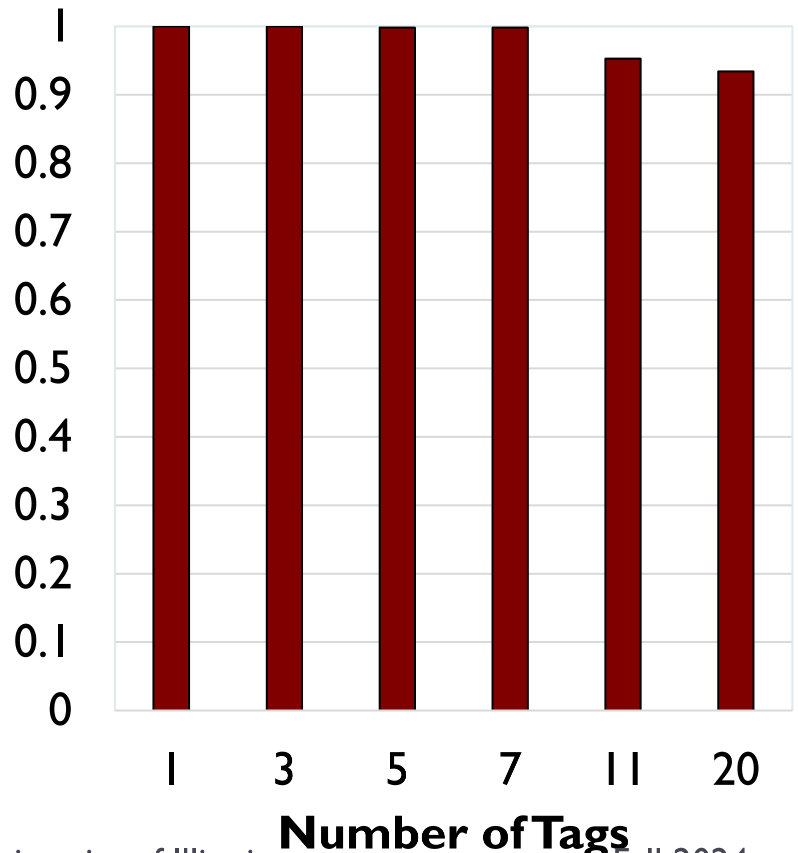
Success = all packets in train

1 sec advertising interval

t = 2.5, 5, 7.5, 10, 15

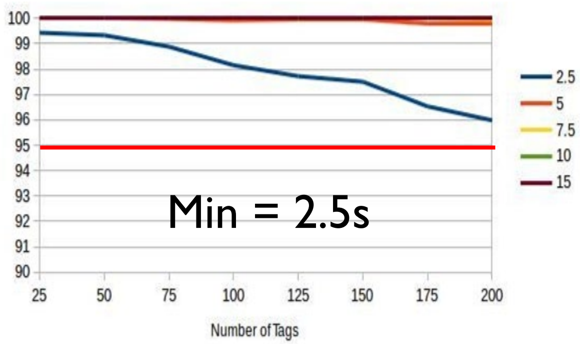
Active Scanning vs. N=2 Beacon Trains (experimental results)



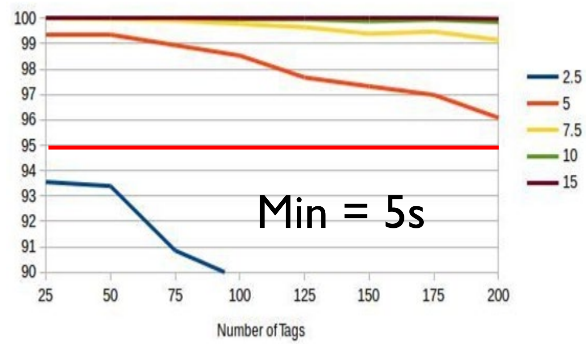


Beacon Train Success - 1 sec BI

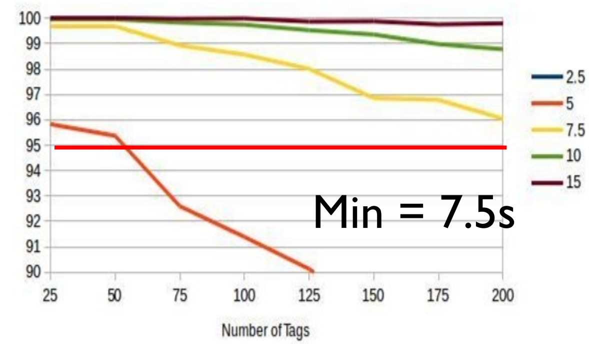
N = 1



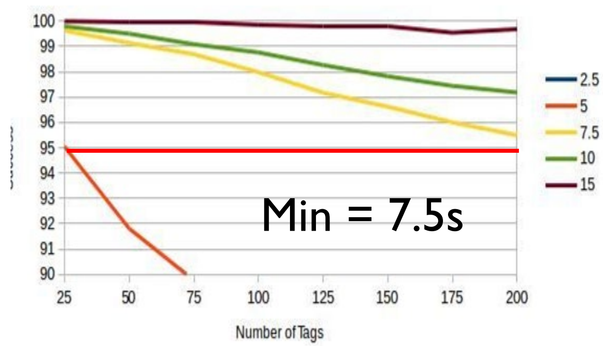
N = 2



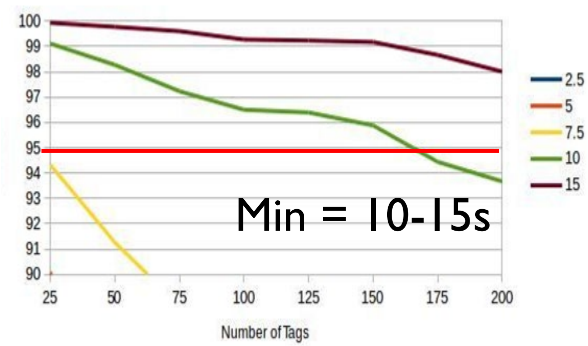
N = 3



N = 4



N = 5



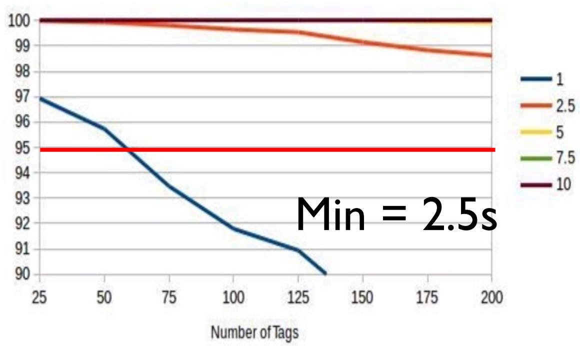
Success = all packets in train

1 sec advertising interval

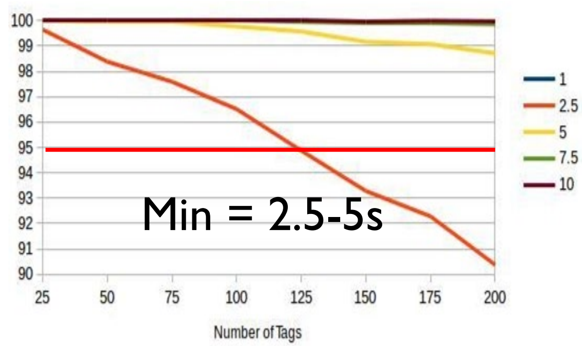
t = 2.5, 5, 7.5, 10, 15

Beacon Train Success - 500 msec BI

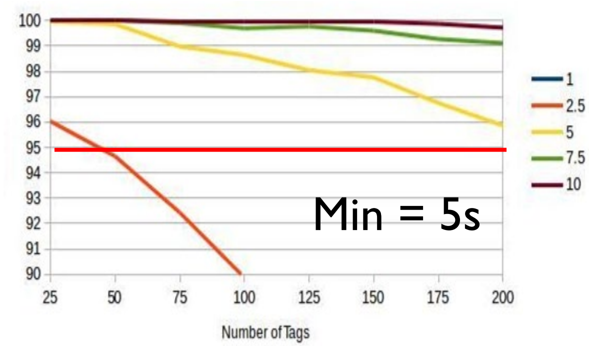
N = 1



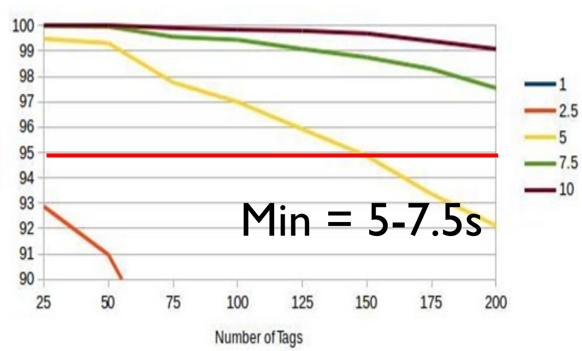
N = 2



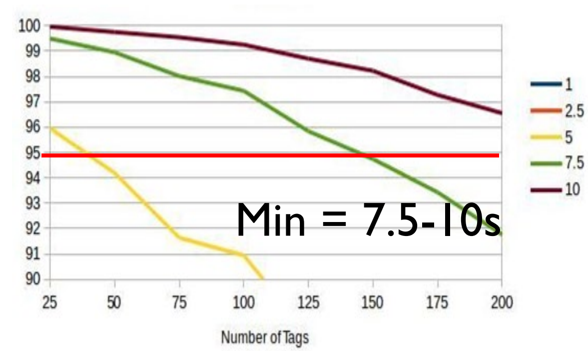
N = 3



N = 4



N = 5



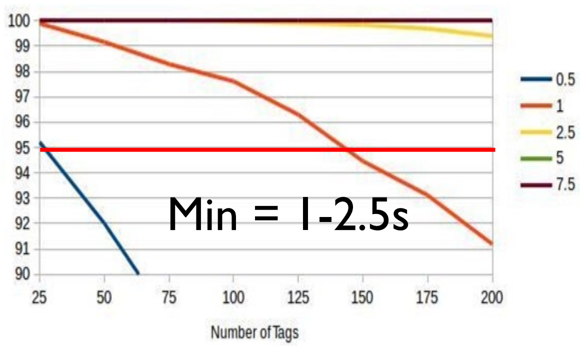
Success = all packets in train

500 msec advertising interval

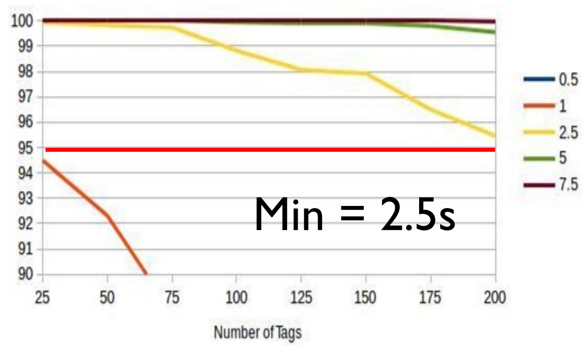
t = 1, 2.5, 5, 7.5, 10

Beacon Train Success - 250 msec BI

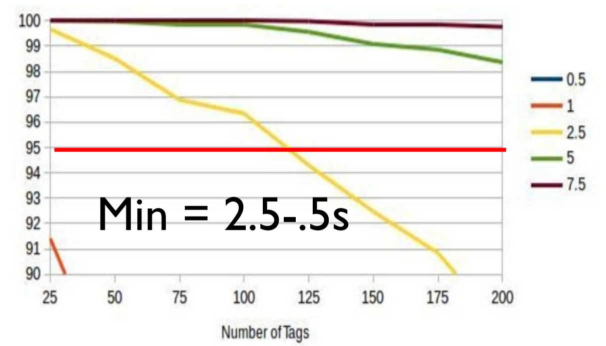
N = 1



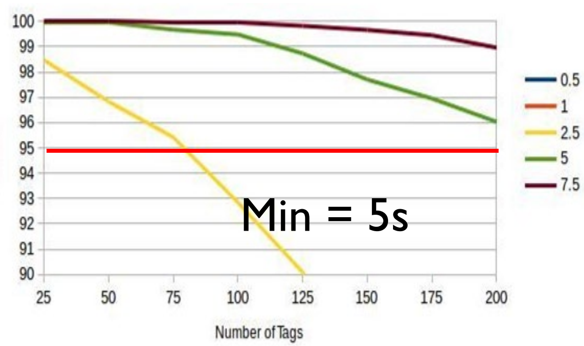
N = 2



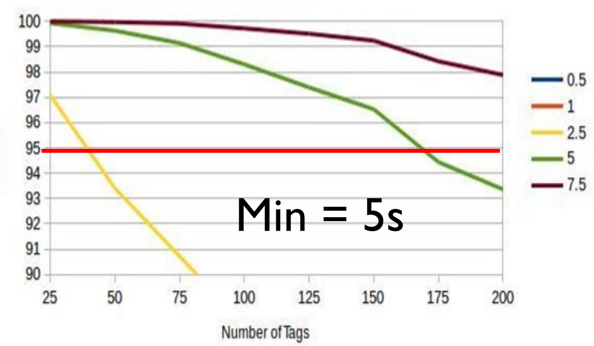
N = 3



N = 4



N = 5

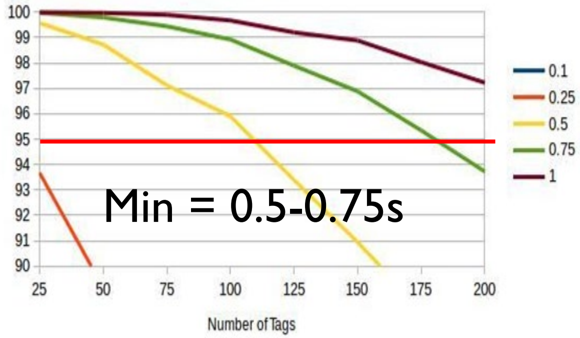


Success = all packets in train

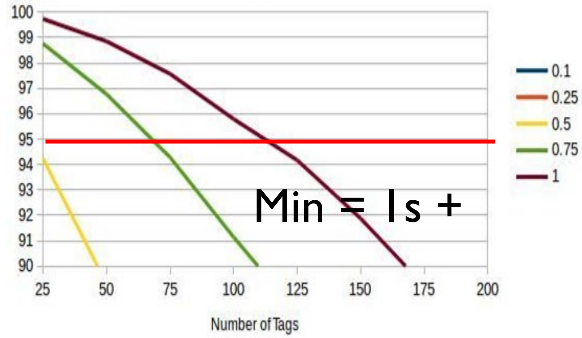
250 msec advertising interval
t = 0.5, 1, 2.5, 5, 7.5

Beacon Train Success - 100 msec BI

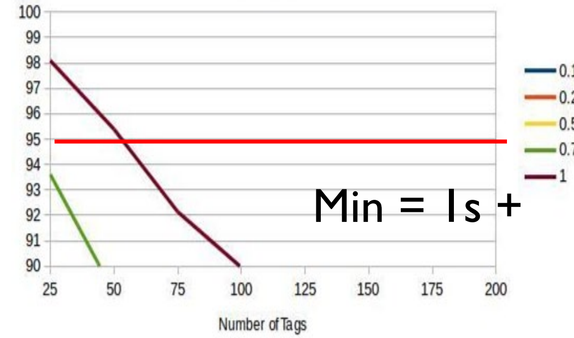
N = 1



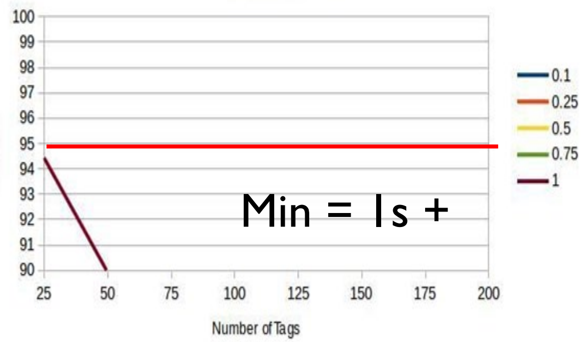
N = 2



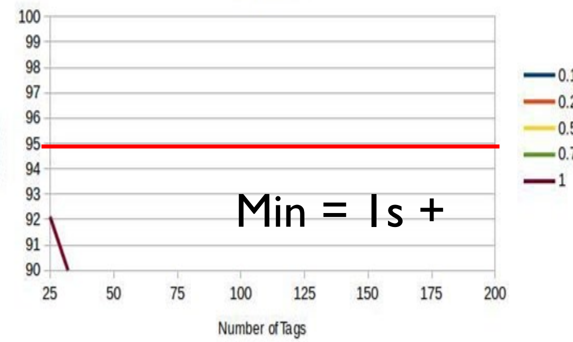
N = 3



N = 4



N = 5



Success = all packets in train

100 msec advertising interval
t = 0.1, 0.5, 1, 2.5, 5