Anonymizing Wireless Discovery

Fall 2024

IoT: The Vision

Connecting users to the information around them …

IoT: The Vision

Connecting users to the information around them …

to enable better recommendations , services and overall user experiences …

IoT: The Vision

Connecting users to the information around them …

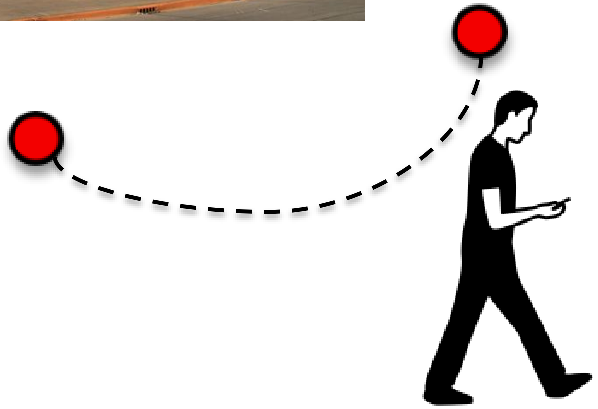
to enable better recommendations , services and overall user experiences …

IoT: The Reality

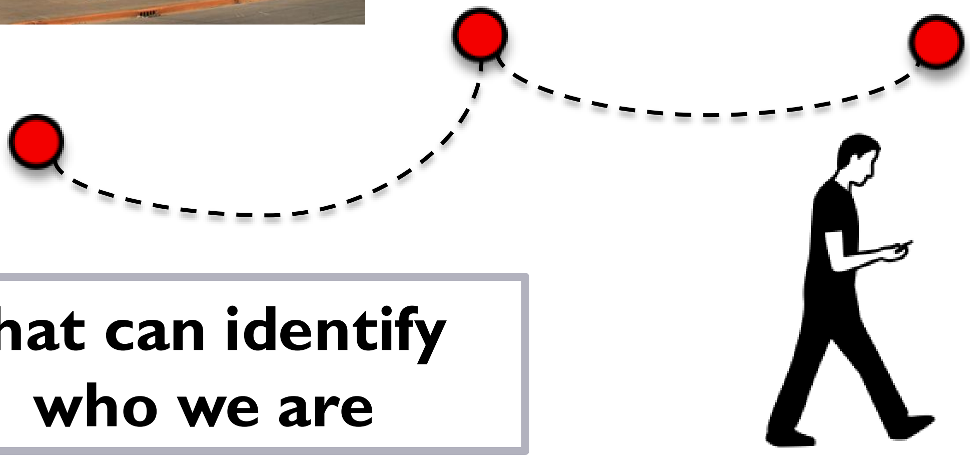
As we move through the world and interact with our environments …

IoT: The Reality

we leave behind breadcrumbs

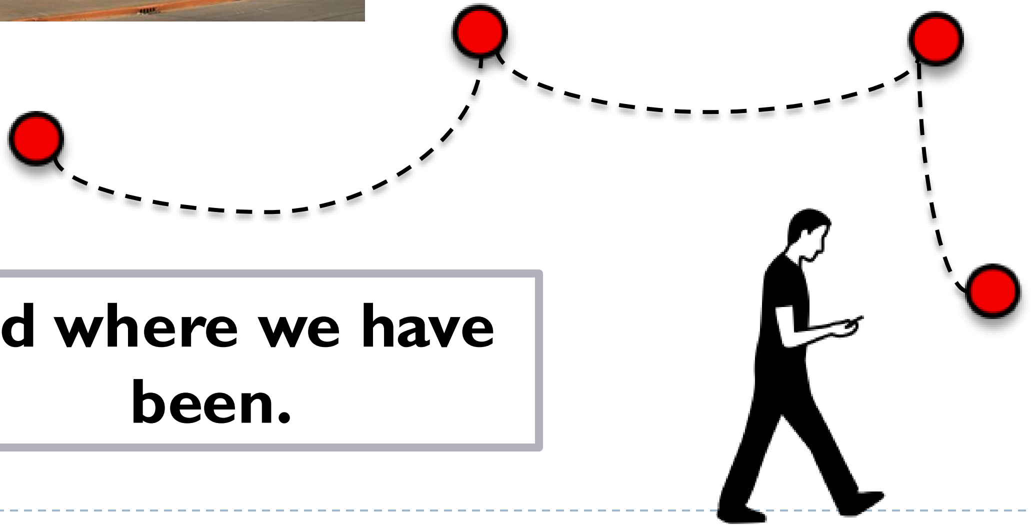


IoT: The Reality



that can identify who we are

IoT: The Reality



and where we have been.

IoT: The Problem

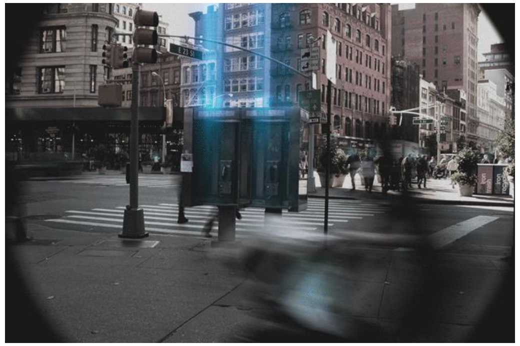
BuzzFeed NEWS

TECH

Exclusive: Hundreds Of Devices Hidden Inside New York City Phone Booths

Beacons can push you ads — and help track your every move. Update: Hours after BuzzFeed News exposed the devices, the city ordered the removal of the devices.

Anyone can now track the user!



IoT: The Problem

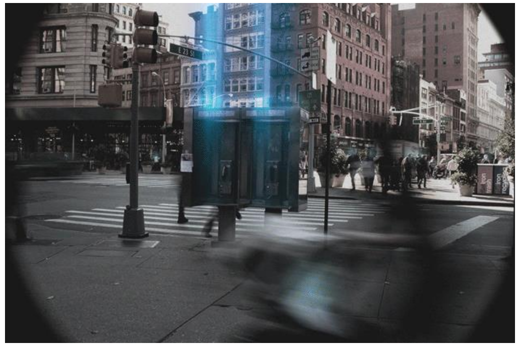
BuzzFeed NEWS

TECH

Exclusive: Hundreds Of Devices Hidden Inside New York City Phone Booths

Beacons can push you ads — and help track your every move. Update: Hours after BuzzFeed News exposed the devices, the city ordered the removal of the devices.

No one should ever use IoT if we can't provide privacy



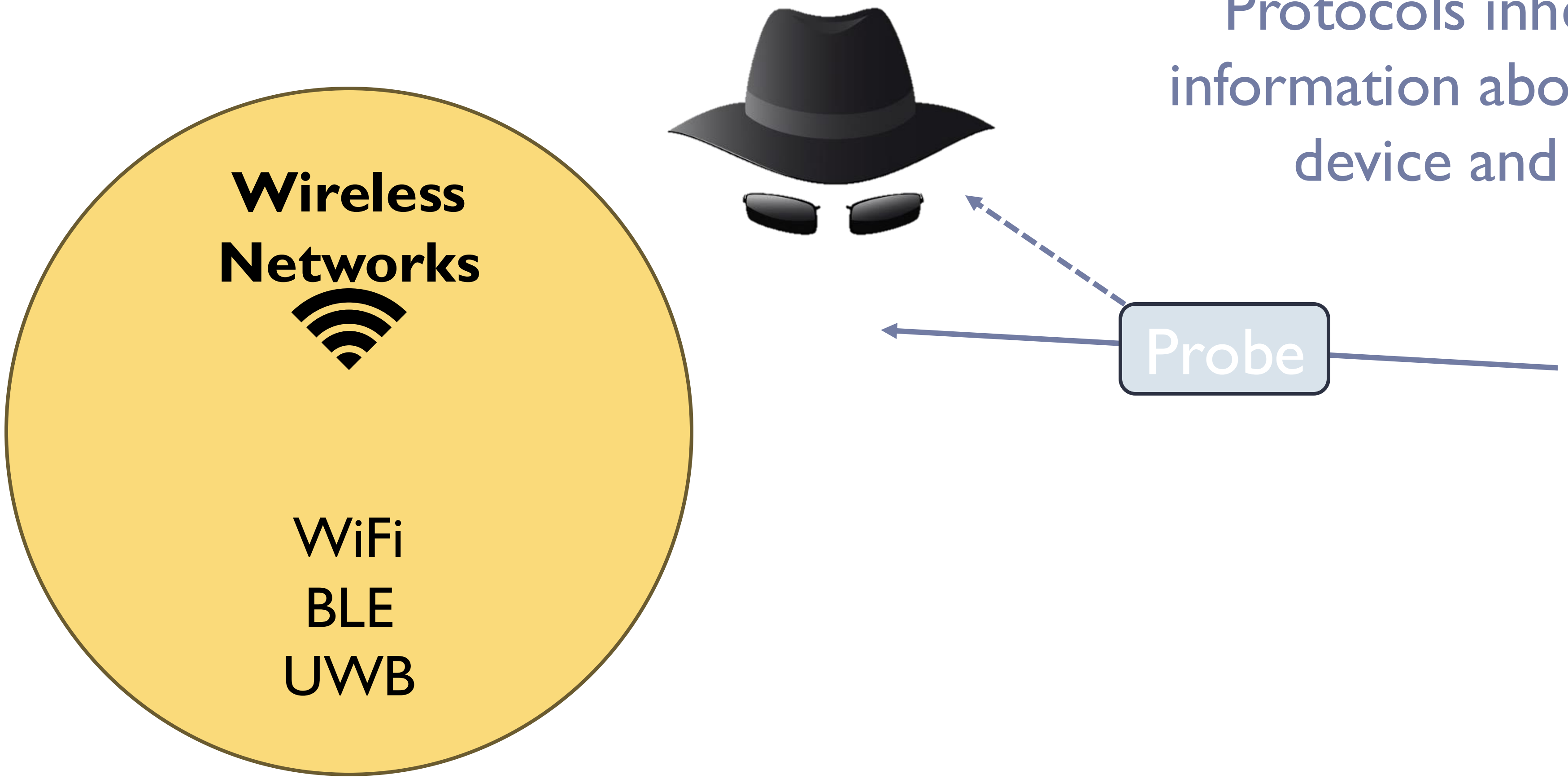
Wireless is Pervasive

Wireless Networks

WiFi
BLE
UWB

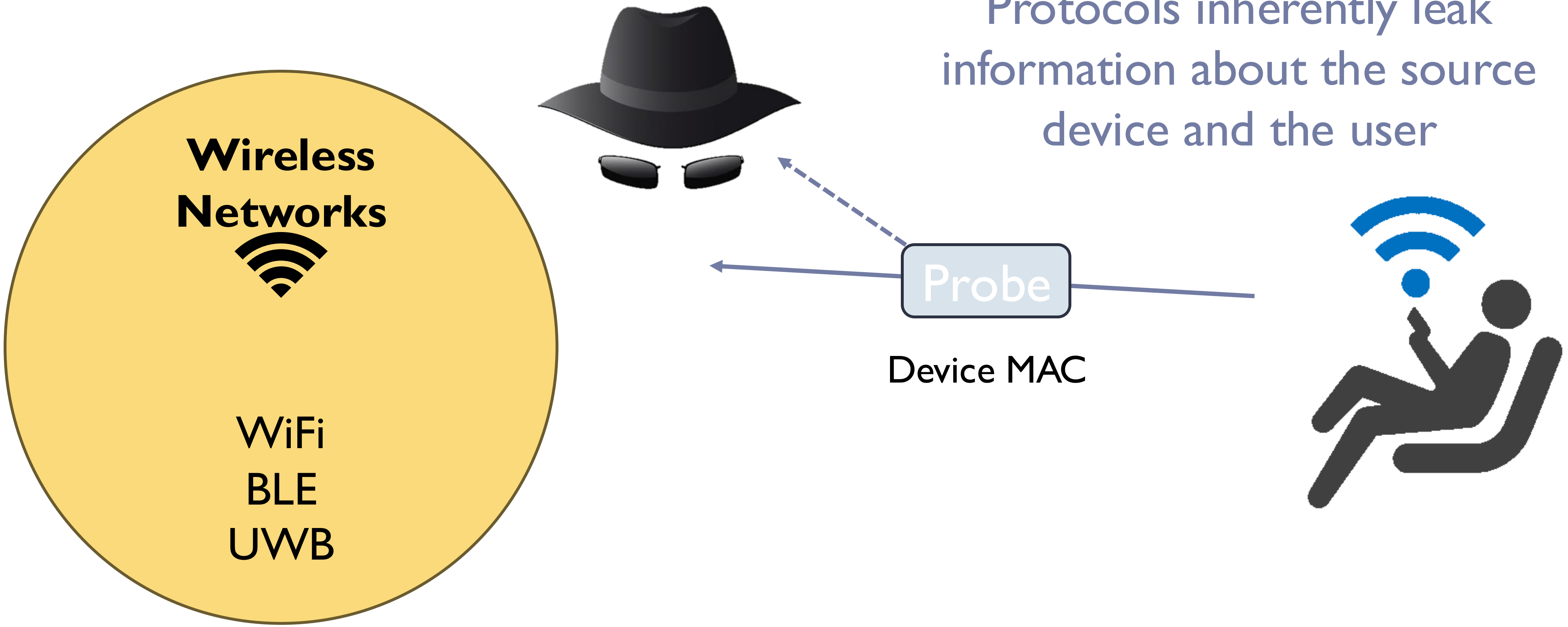
Problem: Wireless is Invasive

Protocols inherently leak information about the source device and the user

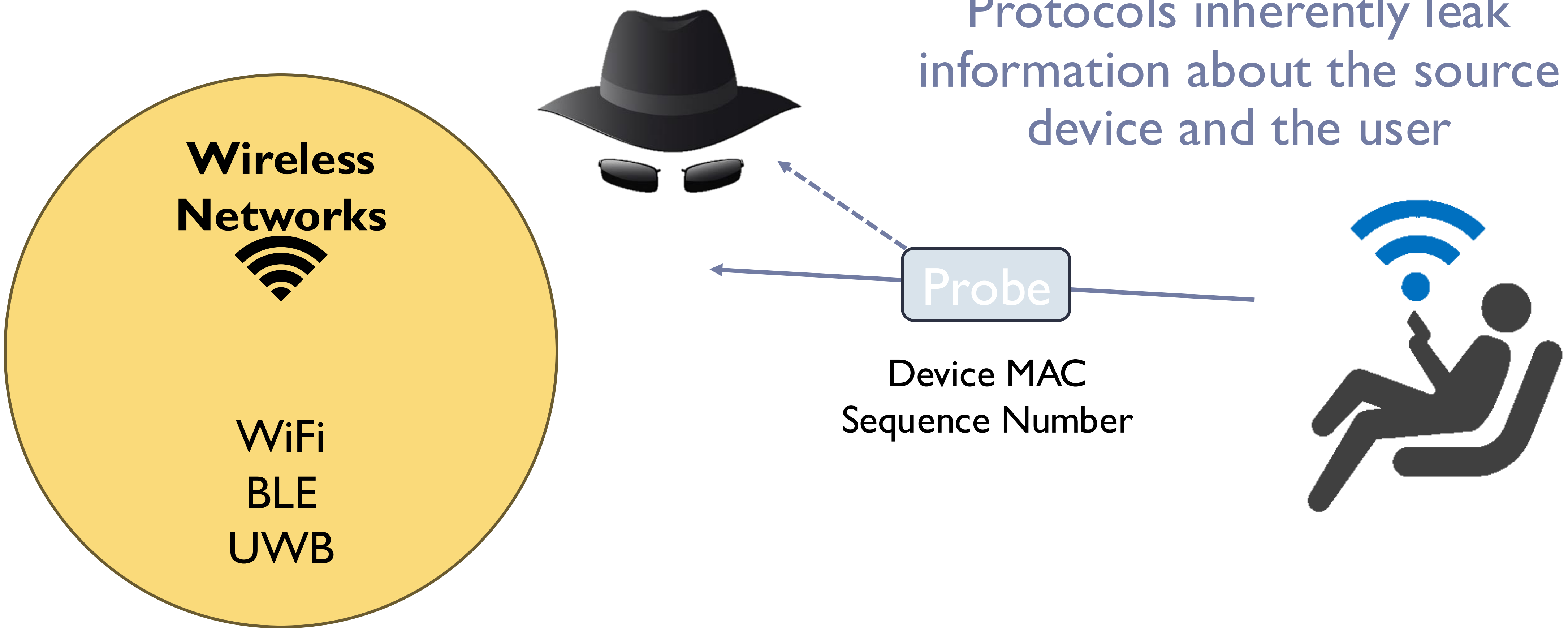


Problem: Wireless is Invasive

Protocols inherently leak information about the source device and the user



Problem: Wireless is Invasive



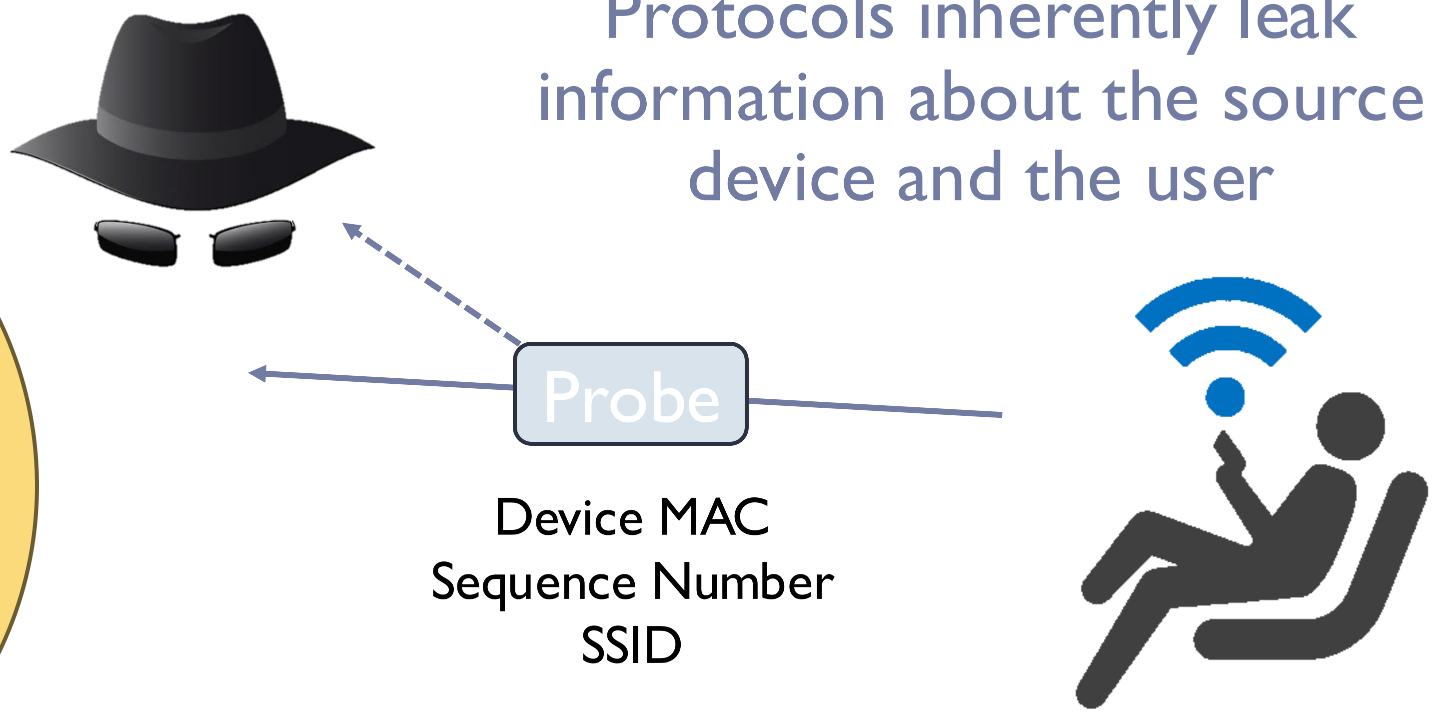
Protocols inherently leak information about the source device and the user

Problem: Wireless is Invasive

Wireless Networks

WiFi
BLE
UWB

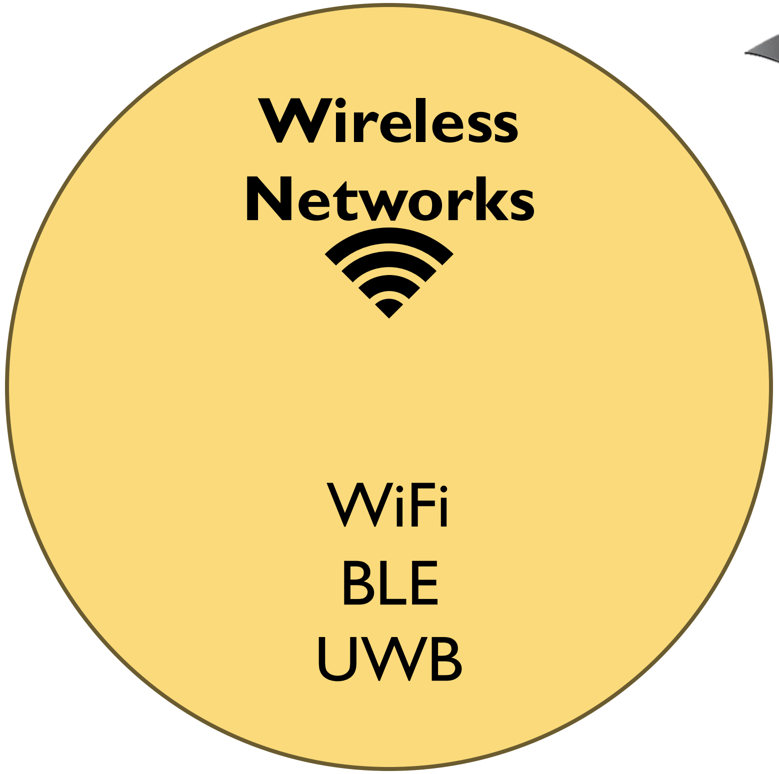
Protocols inherently leak information about the source device and the user



Device MAC
Sequence Number
SSID

02:fe:de:be:ef:f0
IllinoisNet
Starbucks WiFi
My Home Network

Problem: Wireless is Invasive



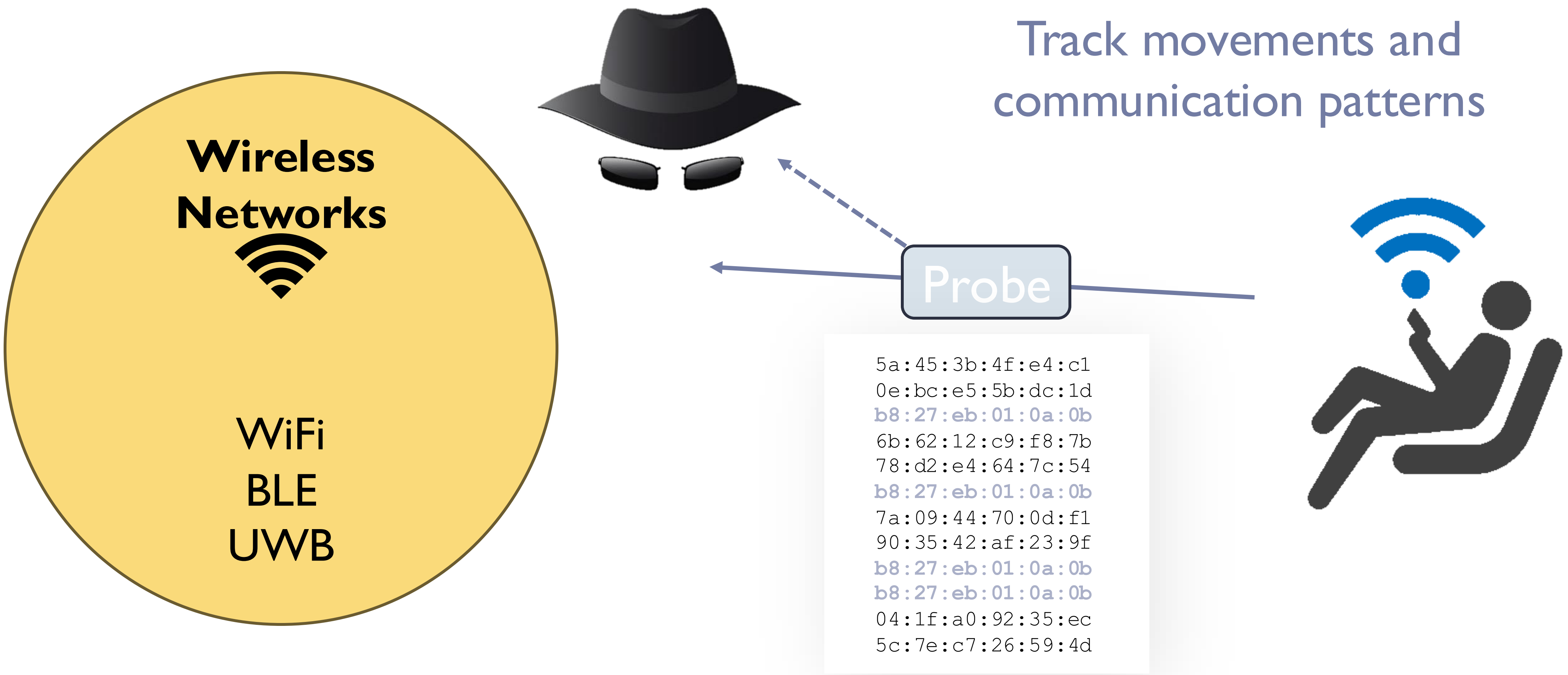
Protocols inherently leak information about the source device and the user

Probe

Device MAC
Sequence Number
SSID
Protocol Parameters

Supported rates

Problem: Wireless is Invasive



Add headline logos

Wireless is Invasive – But Who Cares?

Network Discovery

PCWorld

How 'free' Wi-Fi hotspots can track your location even when you aren't connected

CircleID

Researchers Expose Privacy Risks in Apple and Starlink's Geo-Location Data, Uncovering Military and Civilian Tracking

May 21, 2024, 12:22 pm PDT

ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE STO

BIZ & IT —

No, this isn't a scene from *Minority Report*. This trash can *is* stalking you

Smartphone-monitoring bins in London track places of work, past behavior, and more.

DAN GOODIN - 8/9/2013, 1:15 PM

Communication

Malwarebytes LABS

City fined for tracking its citizens via their phones

Posted: April 29, 2021 by Pieter Arntz

cybernews®

Anyone can tap into your WiFi location data to track you, researchers find

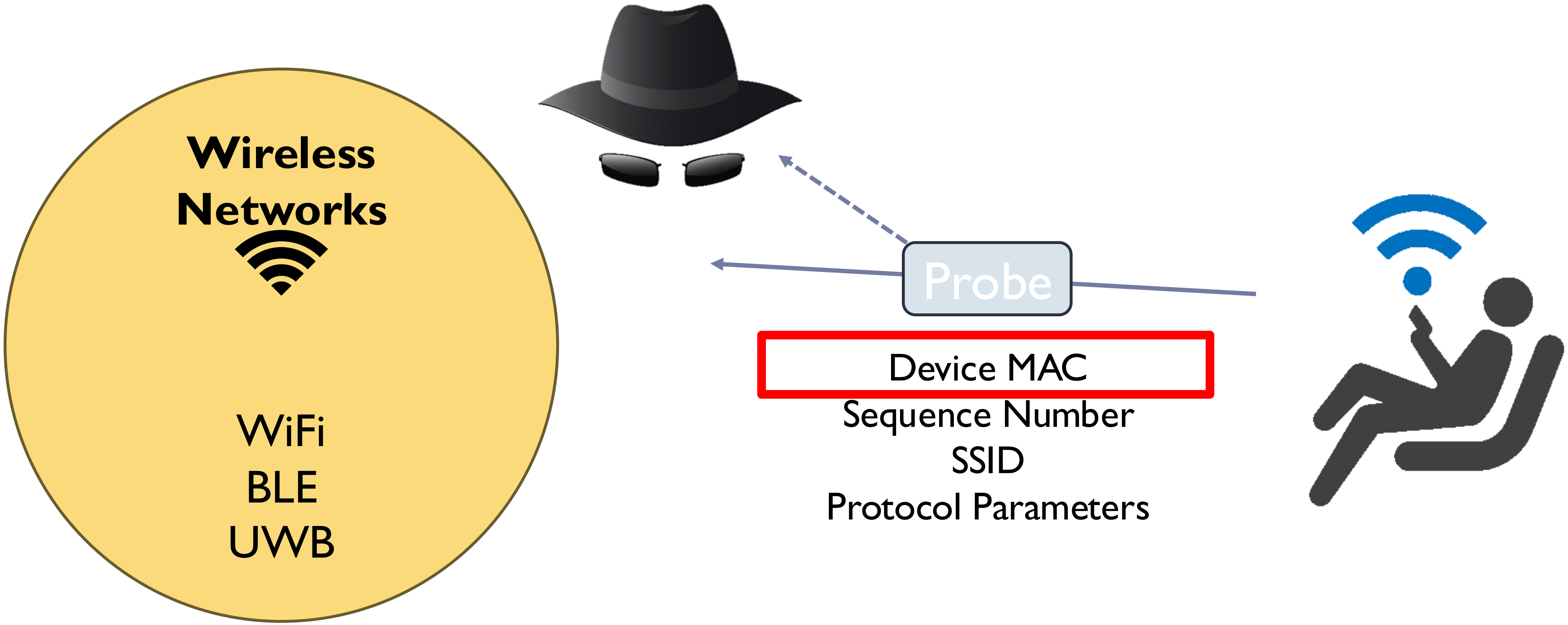
Updated on: May 29, 2024 6:44 AM

Ernestas Naprys, Senior Journalist

ACLU About

Wi-Fi is Another Way We Can Be Tracked 24/7

Problem: Wireless is Invasive

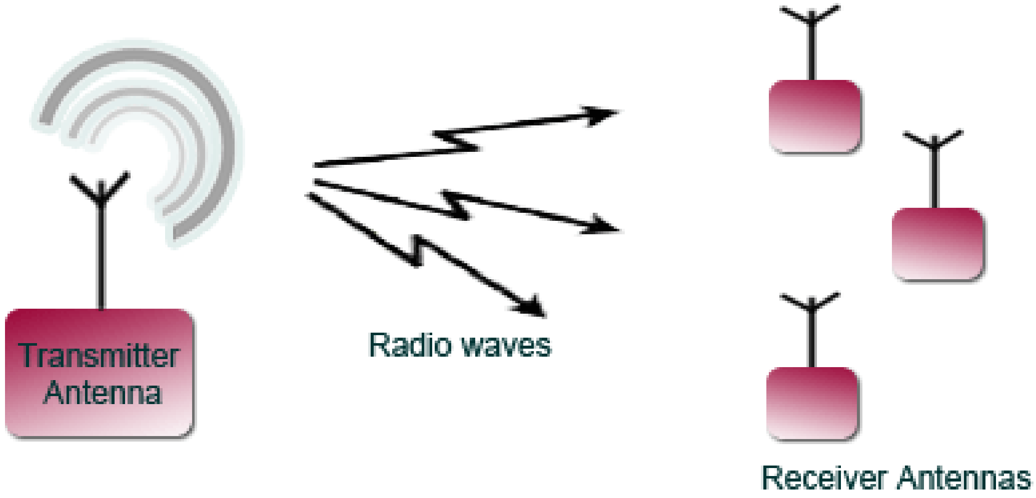


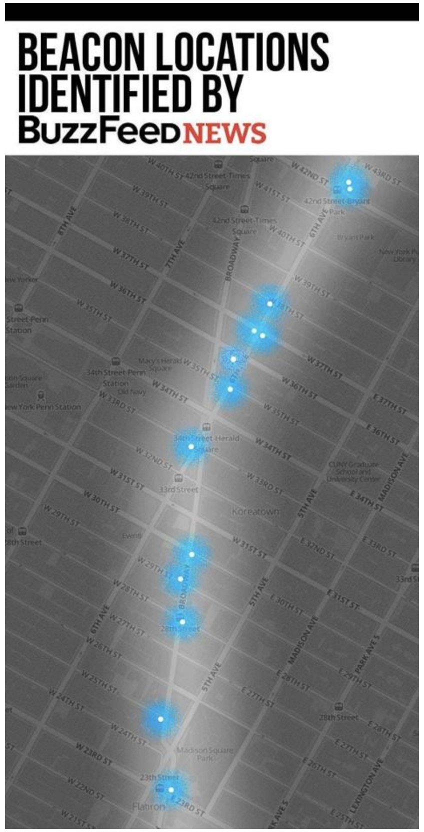
Anonymizing Discovery

- MAC Randomization
  - Change device MAC address from the factory-assigned address
    - WiFi: Discovery
    - BLE: Advertising
  - Enabled by default on most devices
    - Found in mobile OSes from Apple, Android, Windows, Samsung

Managing BLE Exposure

All transmissions contain the identity of the sender (MAC address)





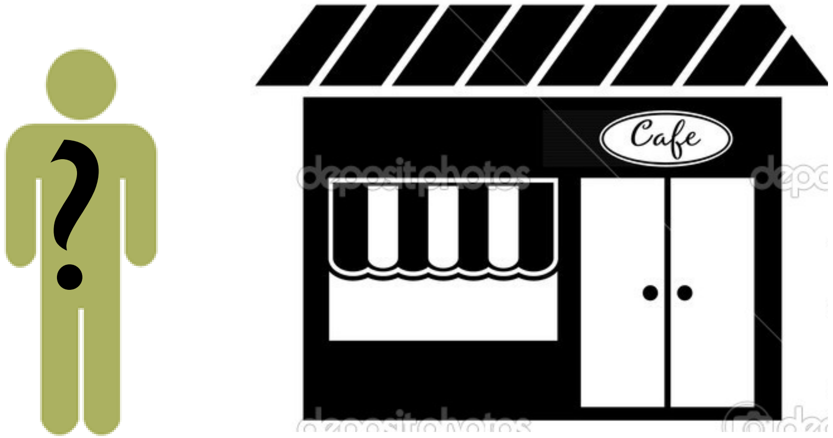
Anyone can listen and track the user

Managing BLE Exposure

Cycle though random MAC address

↓

No one knows who you are!

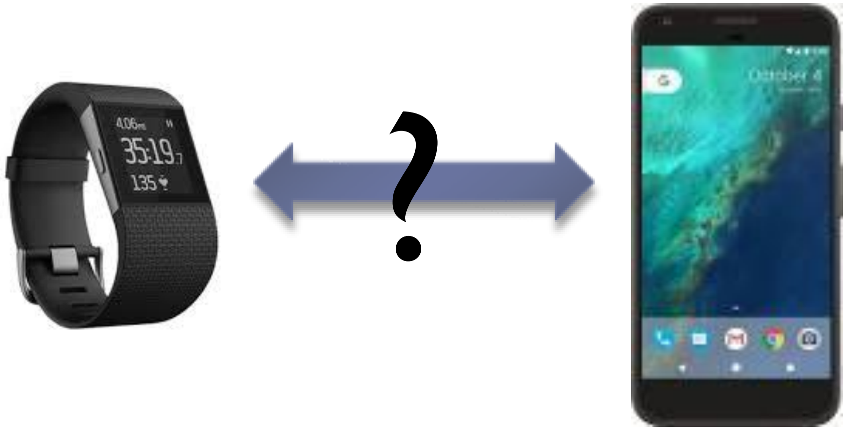


Managing BLE Exposure

Not even your own devices!

Cycle though random MAC address

No one knows who you are!



Managing BLE Exposure

Cycle though random MAC address

No one knows who you are!

Add cid into beacon message

Uses limited 31B of data every time

Managing BLE Exposure

Cycle though random MAC address

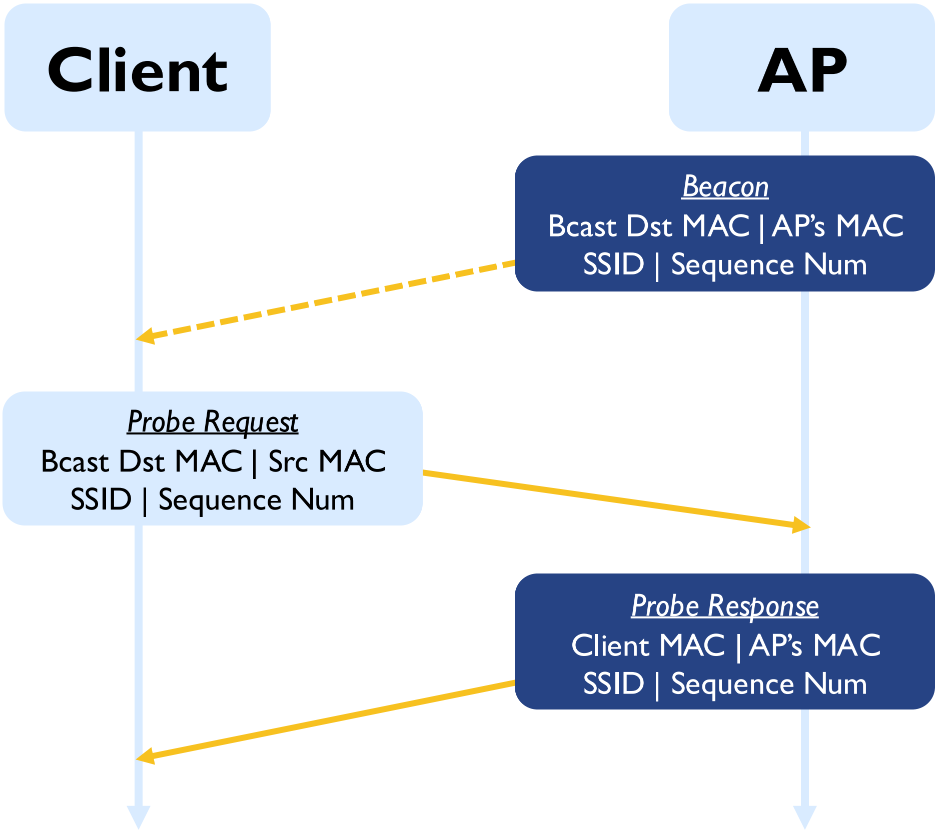
No one knows who you are!

Add cid into beacon message

Uses limited 31B of data every time

Solution: Privacy-preserving MAC addresses

Wi-Fi Discovery



5a:45:3b:4f:e4:c1
0e:bc:e5:5b:dc:1d
b8:27:eb:01:0a:0b
6b:62:12:c9:f8:7b
78:d2:e4:64:7c:54
b8:27:eb:01:0a:0b
7a:09:44:70:0d:f1
90:35:42:af:23:9f
b8:27:eb:01:0a:0b
b8:27:eb:01:0a:0b
04:1f:a0:92:35:ec
5c:7e:c7:26:59:4d

MAC address is kept the same in each probe event

Wi-Fi Discovery with MAC Randomization

5a:45:3b:4f:e4:c1
0e:bc:e5:5b:dc:1d
5c:71:e9:7c:df:5d
6b:62:12:c9:f8:7b
78:d2:e4:64:7c:54
27:19:32:4a:da:e2
7a:09:44:70:0d:f1
90:35:42:af:23:9f
34:20:99:49:ad:8f
ed:4a:75:7d:21:1a
04:1f:a0:92:35:ec
5c:7e:c7:26:59:4d

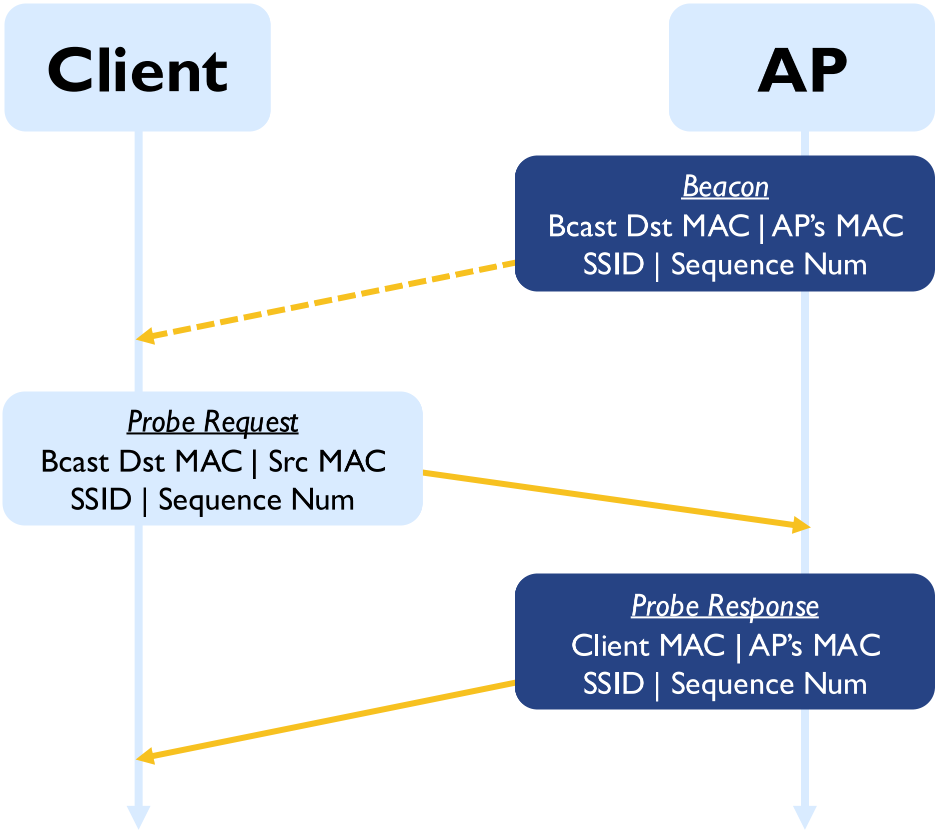
5a:45:3b:4f:e4:c1
0e:bc:e5:5b:dc:1d
b8:27:eb:01:0a:0b
6b:62:12:c9:f8:7b
78:d2:e4:64:7c:54
b8:27:eb:01:0a:0b
7a:09:44:70:0d:f1
90:35:42:af:23:9f
b8:27:eb:01:0a:0b
b8:27:eb:01:0a:0b
04:1f:a0:92:35:ec
5c:7e:c7:26:59:4d

MAC Randomization

- No standard for implementation
  - Address Randomization
    - Implemented by each vendor differently
  - Address Rotation
    - Persistent randomization: use a single random MAC address
    - Non-persistent randomization: use a random MAC address each session
    - Total randomization: use a random MAC address every packet

- Overhead
  - Random MAC address for every packet
    - 6.6% (4ms) overhead on a Raspberry Pi
    - Could be optimized, but is probably overkill

Attacking Wi-Fi Discovery



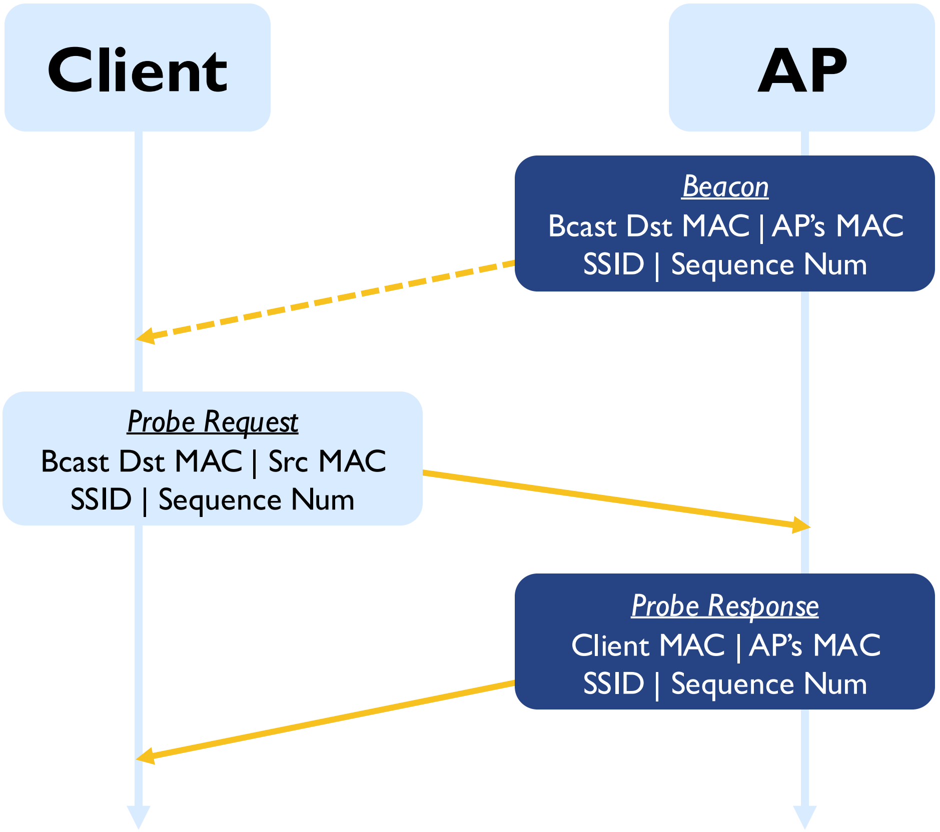
Attacks on Wi-Fi protocols

Control Frame Attack
[Martin et al. PETs '17]

Wi-Peep
[Abedi & Vasisht, MobiCom '22]

Grouping unique Information Elements
[Vanhoef et al. ASIA CCS '16]

Attacking Wi-Fi Discovery



Exploiting leaked info from Wi-Fi

Location Histories
[Han et al. IEEE ICC '18]

Shopping Habits
[Barbera et al. IMC '13]

Users' Workplaces
[Di Luzio et al. INFOCOM '16]

Is MAC Randomization Enough?

Wi-Fi discovery is vulnerable even with MAC randomization

Packet Fields

- MAC Address [Martin et al. PETs '17]
- SSIDs [Han et al. IEEE ICC '18] [Barbera et al. IMC '13]
- Sequence Numbers [Fenske et al. PETs '21] [Freudiger, WiSec '15]

Signal Properties

- Angle of Arrival [Xiong & Jamieson, MobiCom '13]
- Signal strength [Bauer et al. PETs '09]
- Time of Flight [Abedi & Vasisht, MobiCom '22]

Is MAC Randomization Enough?

Wi-Fi discovery is vulnerable even with MAC randomization

Packet Fields

- MAC Address [Martin et al. PETs '17]
- SSIDs [Han et al. IEEE ICC '18] [Barbera et al. IMC '13]
- Sequence Numbers [Fenske et al. PETs '21] [Freudiger, WiSec '15]

Signal Properties

- Angle of Arrival [Xiong & Jamieson, MobiCom '13]
- Signal strength [Bauer et al. PETs '09]
- Time of Flight [Abedi & Vasisht, MobiCom '22]

Protocol Behaviors

- Transmission Timing [Matte et al. WiSec '16]
- Frequency of MAC Randomization [Fenske et al. PETs '21]

Is MAC Randomization Enough?

Wi-Fi discovery is vulnerable even with MAC randomization

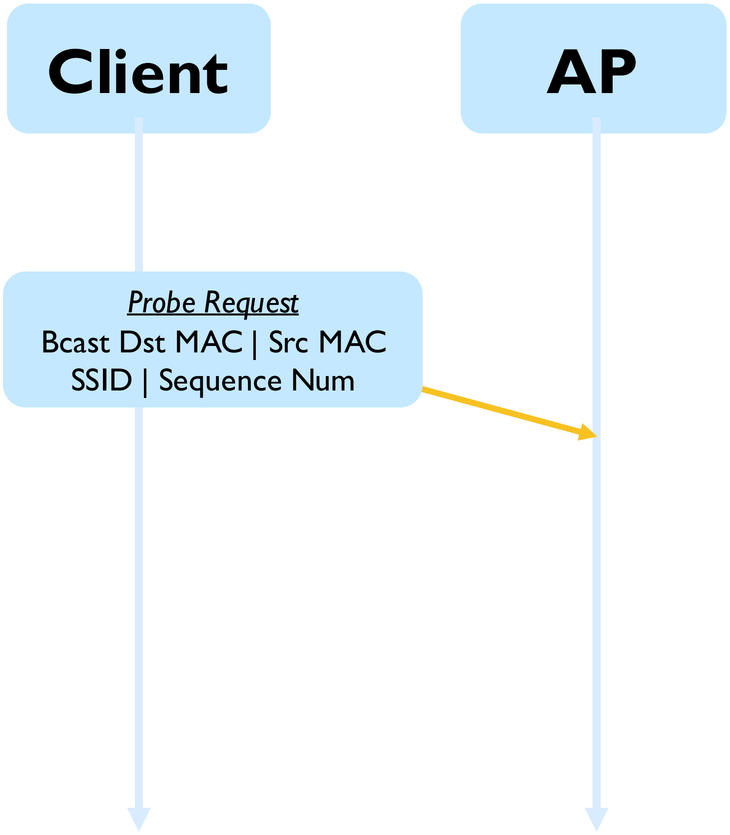
Timing attacks on network discovery

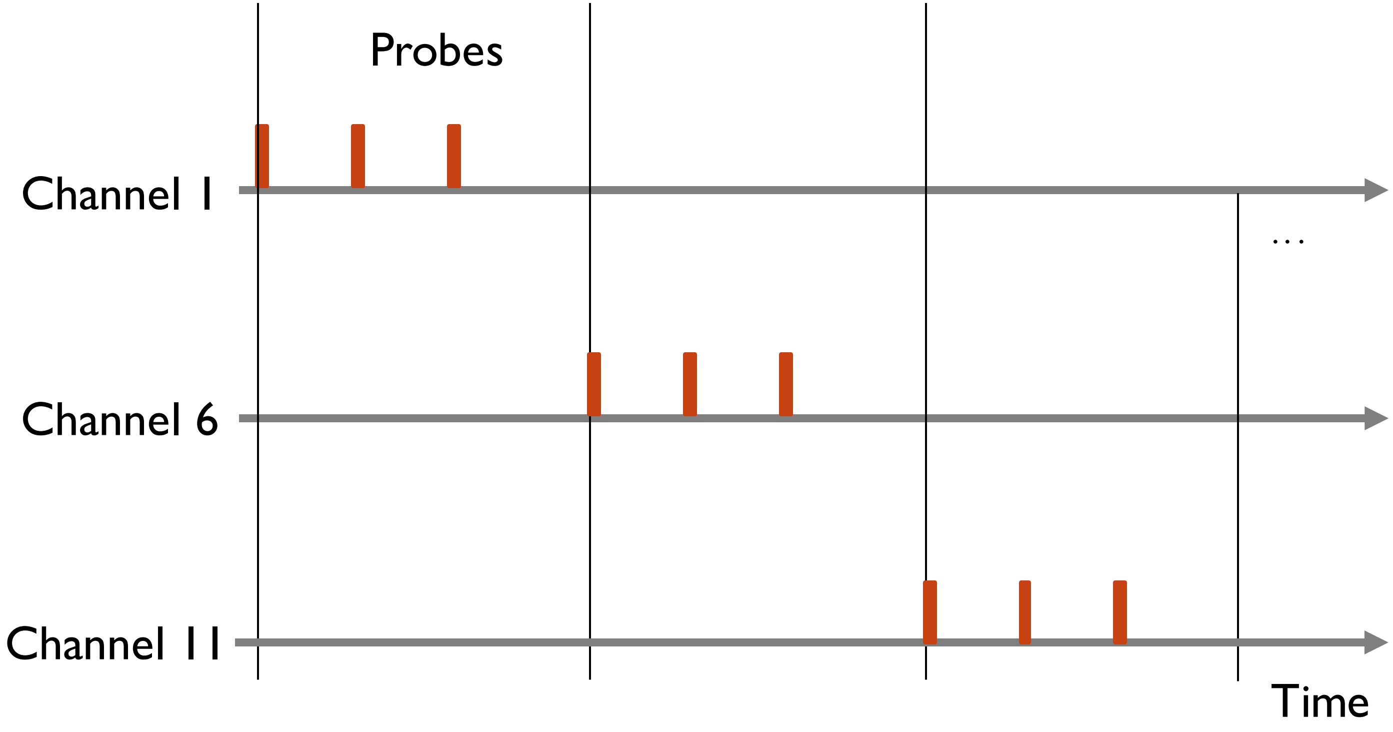
Protocol Behaviors

Transmission Timing
[Matte et al. WiSec '16]

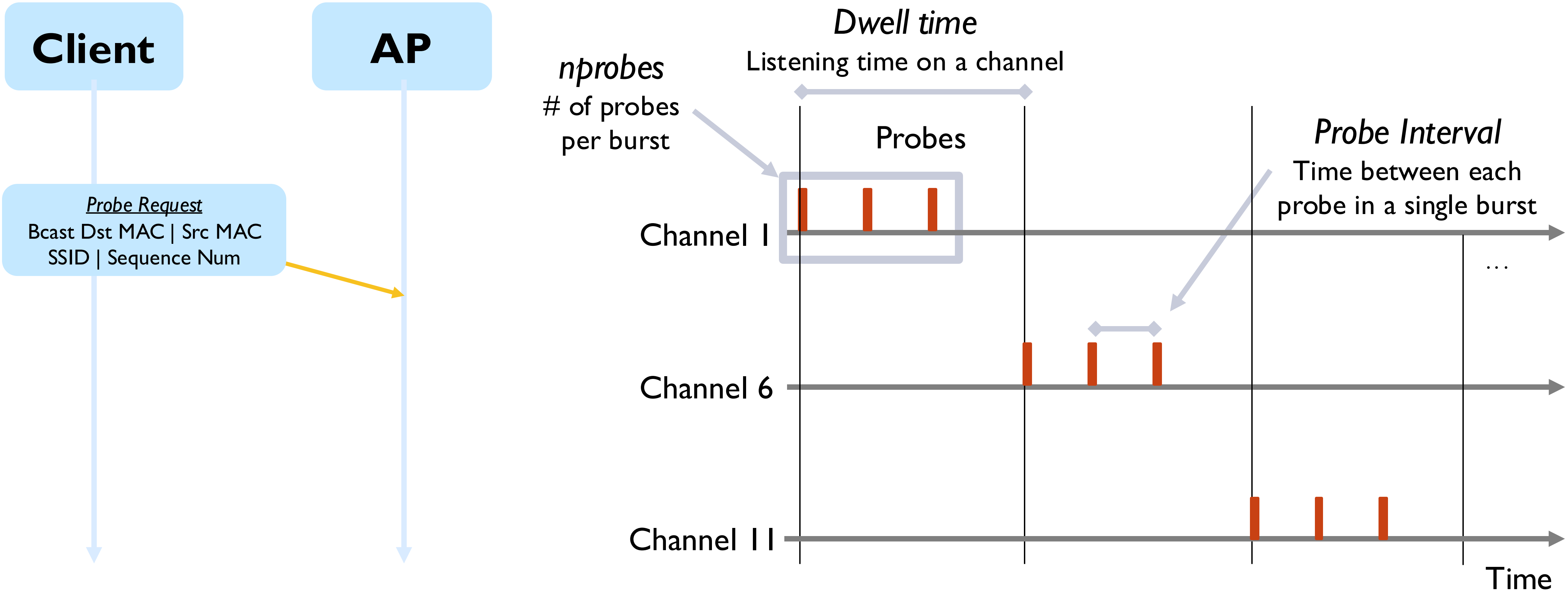
Frequency of MAC Randomization
[Fenske et al. PETs '21]

Network Discovery: Probe Events





Network Discovery: Probe Events



Observed Probe Intervals of Mobile Devices

Device Model	OS Version	Probe Interval
Apple iPhone 14 Pro Max	17.1	20.3ms ± 0.1ms
Apple iPhone 13	16.7.1	20.2ms
Apple iPhone 11	17.0.3	20.2ms ±0.1ms
Apple iPhone SE (2nd gen)	16.6.1	20.2ms± 0.1ms
Google Pixel 7 Pro	14	20ms ± 1ms
Google Pixel 6a	13	--
Samsung S22 Ultra	13	40ms
Samsung S21	13	40ms ± 2ms
Samsung S10e	12	11ms
Raspberry Pi 3B+	RPi OS 6.1	21ms
Raspberry Pi 4B	Kali 2023.2	20ms ± 1ms
Dell Inspiron 15R	Windows 10 22H2	11ms
Lenovo Yoga 710	Ubuntu 20.04	51ms

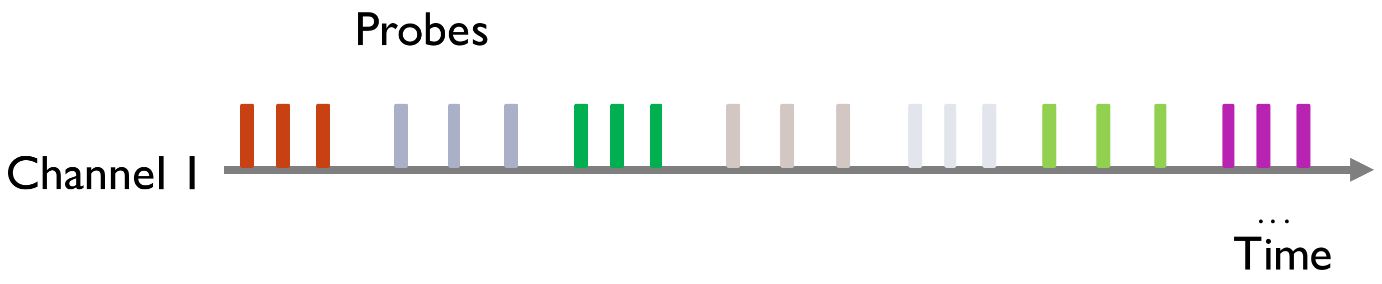
Make table consistent

Exploiting Probe Interval Patterns

Measure the probe intervals, grouped by MAC address

↓

Calculate averages and medians for probe intervals



Transmission Timing
[Matte et al. WiSec '16]

Probe Interval Patterns
[Cifuentes-Urtubey et al. MobiSys '22]

Exploiting Probe Interval Patterns

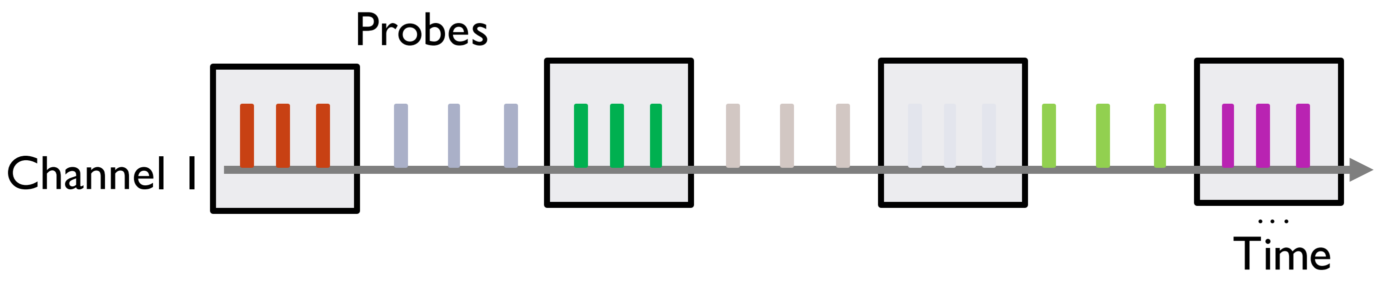
Measure the probe intervals, grouped by MAC address

↓

Calculate averages and medians for probe intervals

↓

Groups with similar stats are considered the same device



Transmission Timing
[Matte et al. WiSec '16]

Probe Interval Patterns
[Cifuentes-Urtubey et al. MobiSys '22]

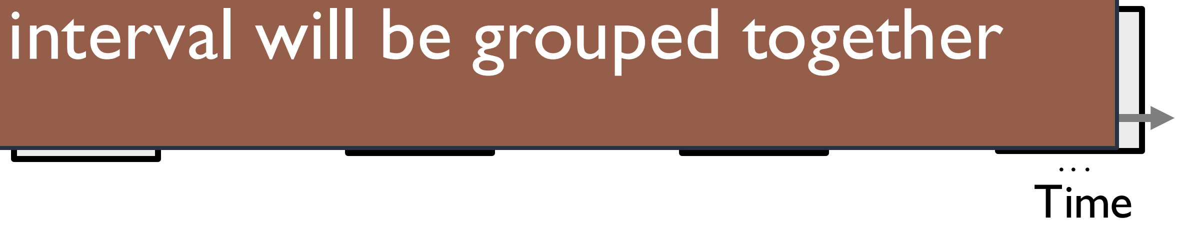
Exploiting Probe Interval Patterns

Measure the probe intervals,

All devices with the same probe interval will be grouped together

for probe intervals

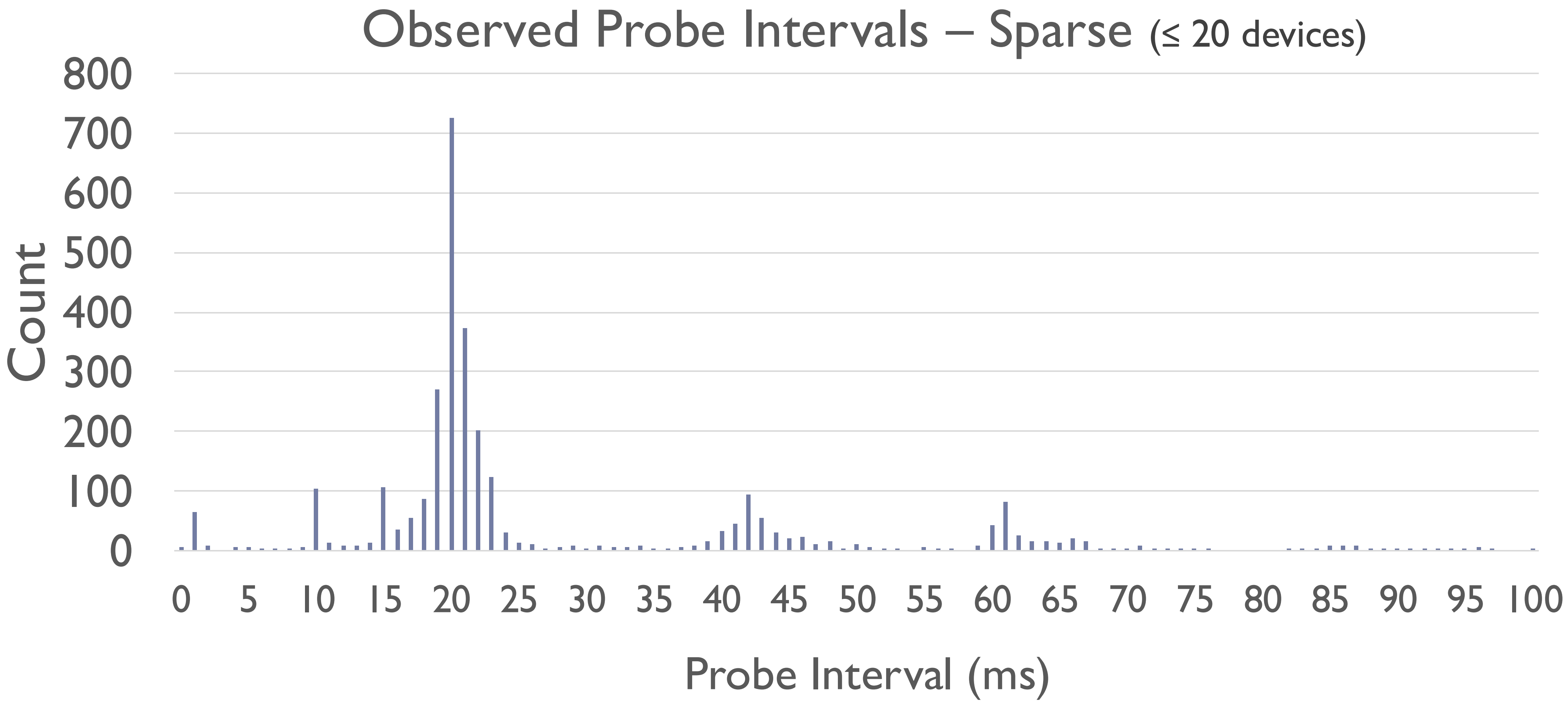
Groups with similar stats are considered the same device



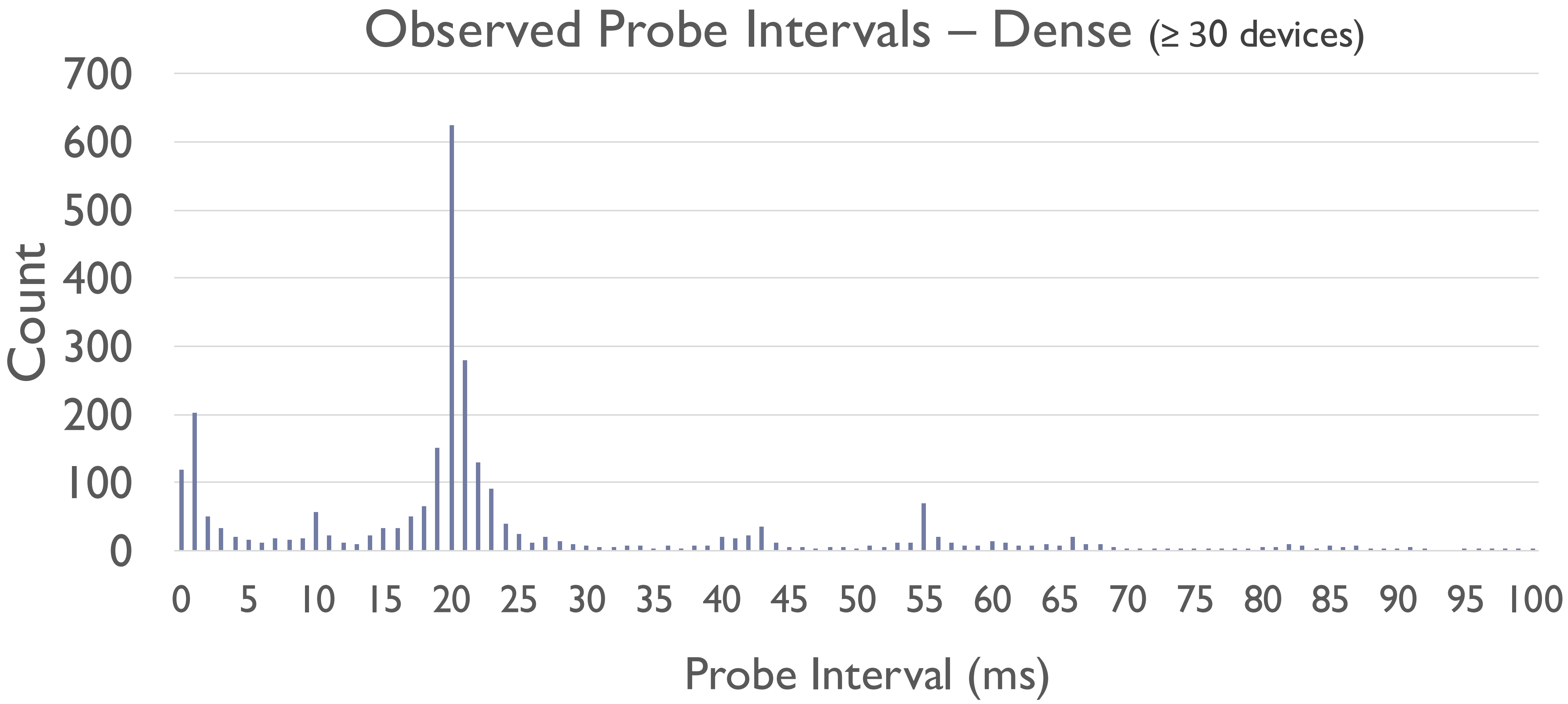
Transmission Timing
[Matte et al. WiSec '16]

Probe Interval Patterns
[Cifuentes-Urtubey et al. MobiSys '22]

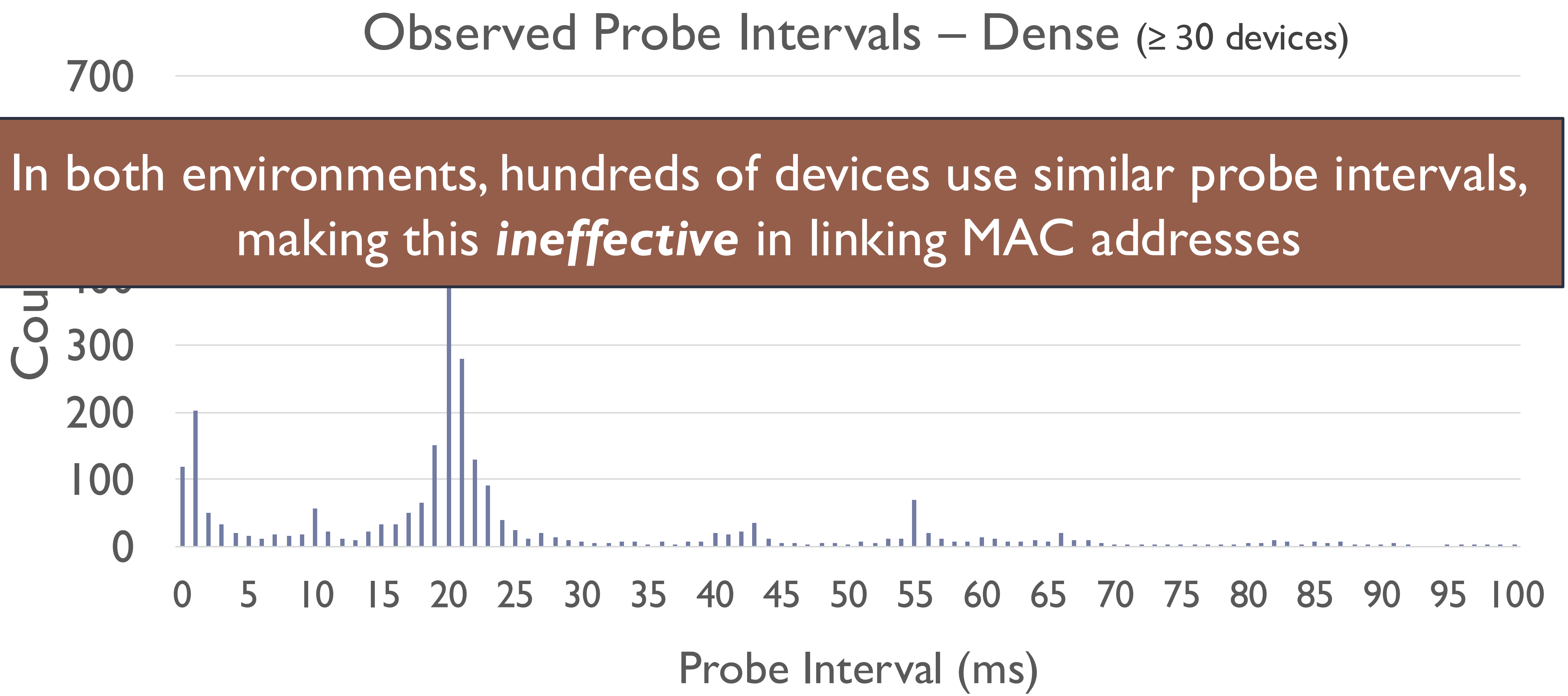
Limitation: Probe Interval Patterns



Limitation: Probe Interval Patterns



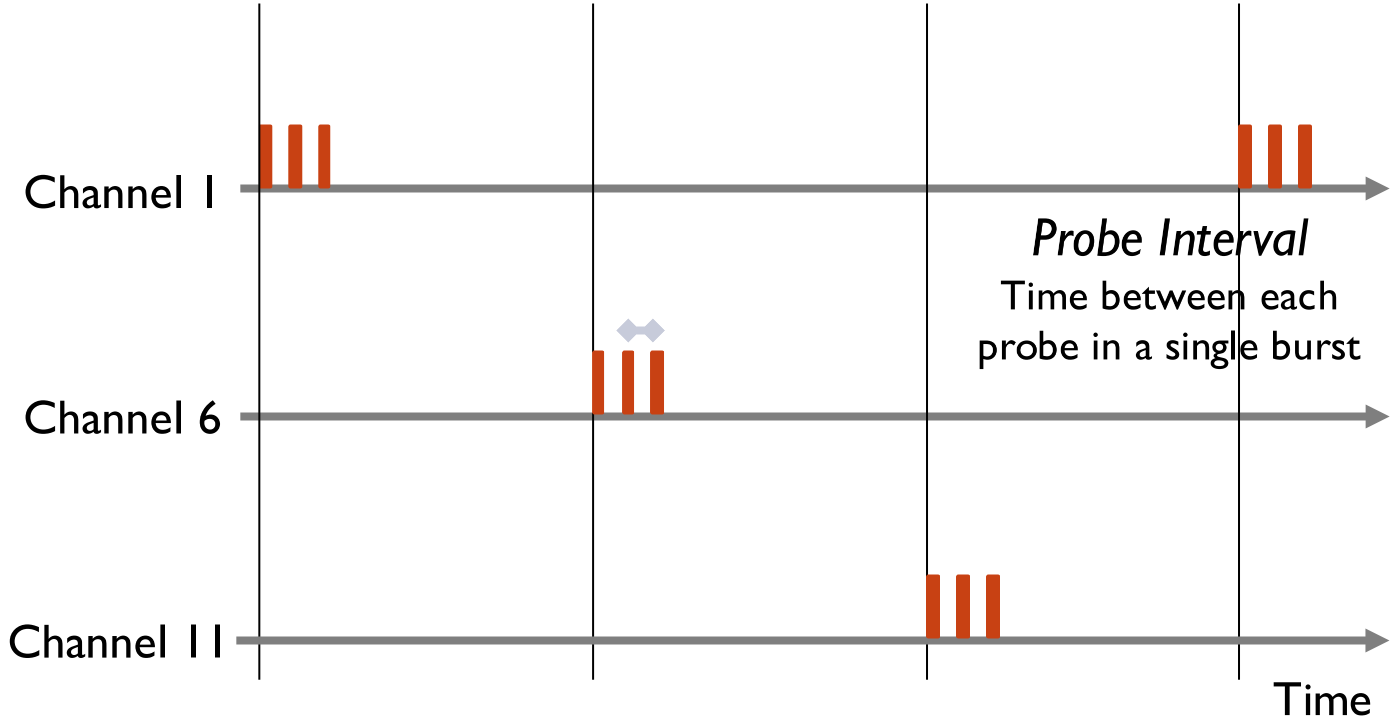
Limitation: Probe Interval Patterns



In both environments, hundreds of devices use similar probe intervals, making this ***ineffective*** in linking MAC addresses

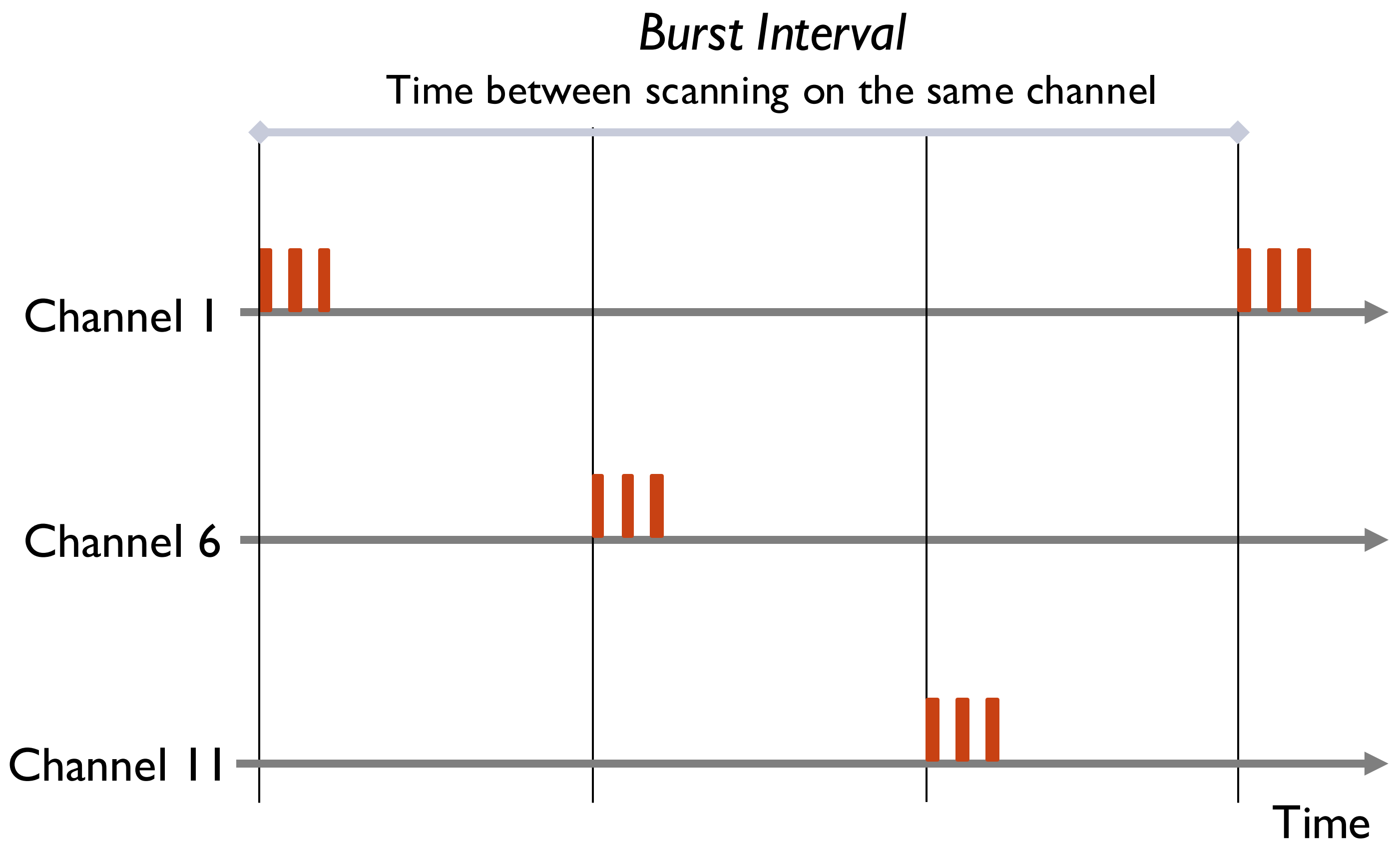
Time Scale is the Key

- Prior work focused solely on probe intervals



Time Scale is the Key

- Prior work focused solely on probe intervals
- New approach: Analyze timing patterns **across bursts**



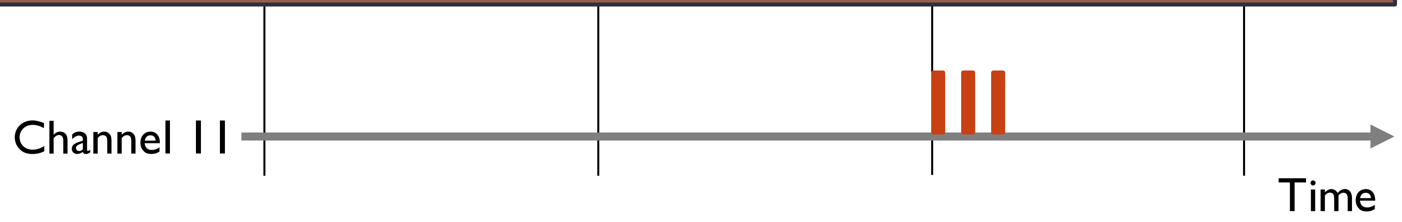
Time Scale is the Key

- Prior work focused

Burst Interval
Time between scanning on the same channel

Device probe events last ~100ms
Devices burst on the order of 10s – 100s of seconds

Probability that probe events from different devices overlap is very low



Observed Burst Intervals

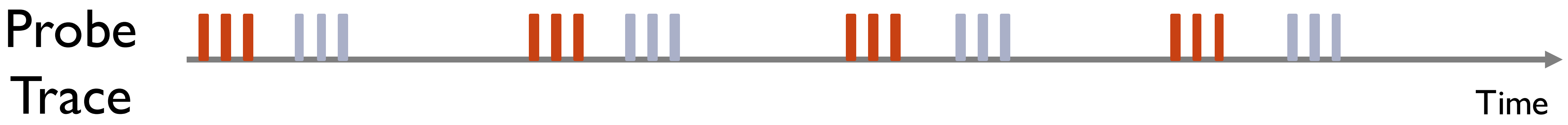
Device Model	OS Version	Probe Interval	Burst Interval
Apple iPhone 14 Pro Max	17.1	20.3ms ± 0.1ms	--
Apple iPhone 13	16.7.1	20.2ms	--
Apple iPhone 11	17.0.3	20.2ms ±0.1ms	--
Apple iPhone SE (2nd gen)	16.6.1	20.2ms± 0.1ms	--
Google Pixel 7 Pro	14	20ms ± 1ms	160 sec
Google Pixel 6a	13	--	160 sec
Samsung S22 Ultra	13	40ms	40 sec
Samsung S21	13	40ms ± 2ms	13 sec
Samsung S10e	12	11ms	40 sec
Raspberry Pi 3B+	RPi OS 6.1	21ms	60sec ± 25ms
Raspberry Pi 4B	Kali 2023.2	20ms ± 1ms	60 sec
Dell Inspiron 15R	Windows 10 22H2	11ms	59.7sec ± 20ms
Lenovo Yoga 710	Ubuntu 20.04	51ms	63.0sec ±30ms

Observed Burst Intervals

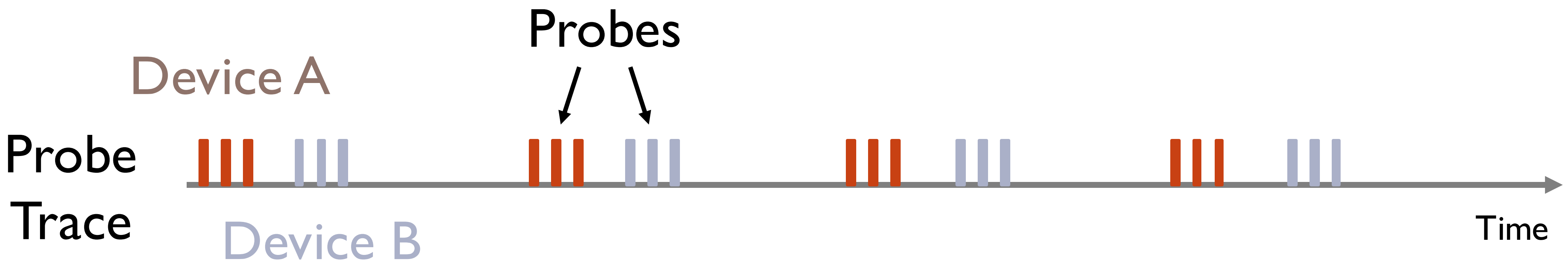
Device Model	OS Version	Probe Interval	Burst Interval
Apple iPhone 14 Pro Max	17.1	20.3ms ± 0.1ms	--
Apple iPhone 13	16.7.1	20.2ms	--
Apple iPhone 11	17.0.3	20.2ms ±0.1ms	--
Apple iPhone SE (2nd gen)	16.6.1	20.2ms± 0.1ms	--
Google Pixel 7 Pro			160 sec
Google Pixel 6a			160 sec
Samsung S22 Ultra			40 sec
Samsung S21			13 sec
Samsung S10e			40 sec
Raspberry Pi 3B+			60sec ± 25ms
Raspberry Pi 4B			60 sec
Dell Inspiron 15R	Windows 10 22H2	11ms	59.7sec ± 20ms
Lenovo Yoga 710	Ubuntu 20.04	51ms	63.0sec ±30ms

Knowing the target burst interval enables tracking the device

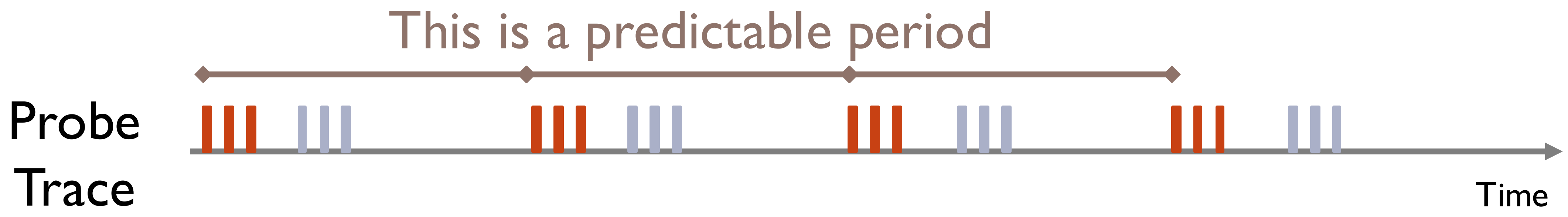
Exploiting Burst Interval Patterns



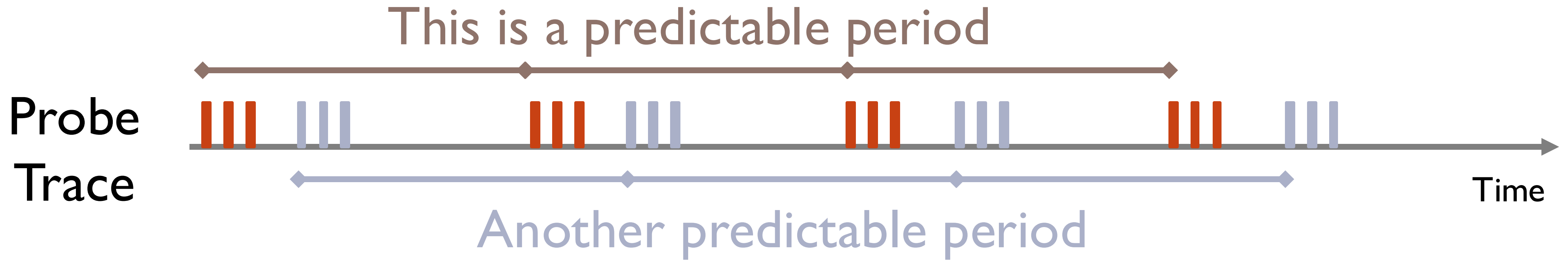
Exploiting Burst Interval Patterns



Exploiting Burst Interval Patterns



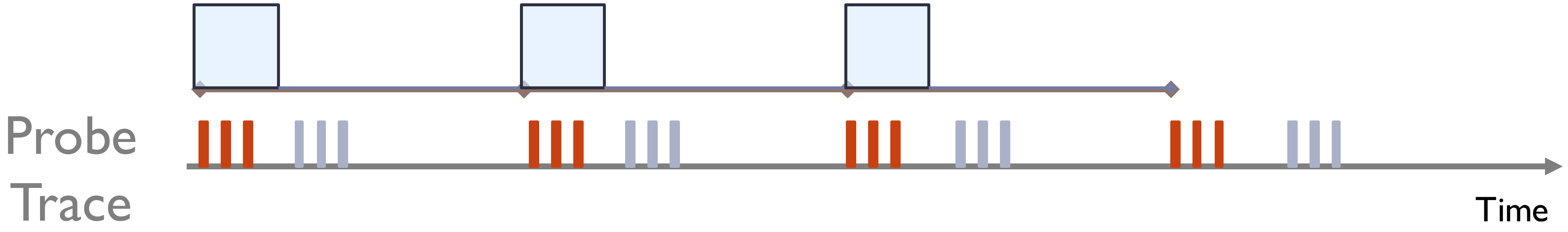
Exploiting Burst Interval Patterns



How do we extract the MAC addresses?

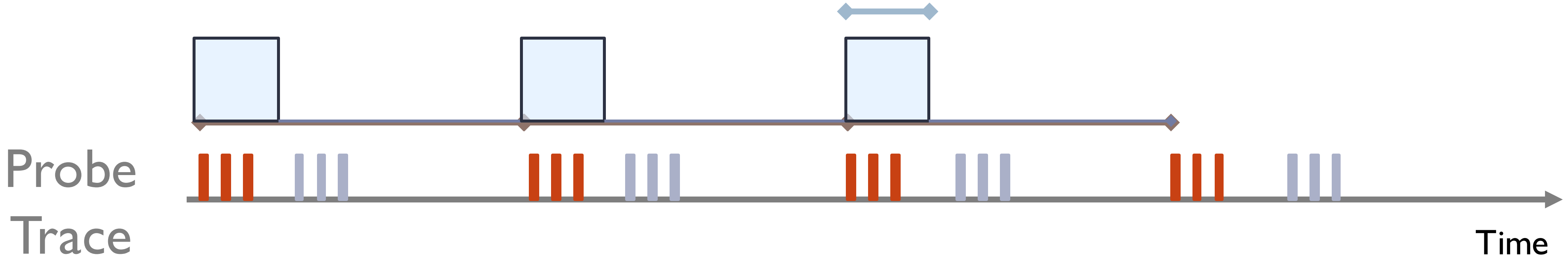
Exploiting Burst Interval Patterns

Create a template (base) pattern of where the probes will be



Exploiting Burst Interval Patterns

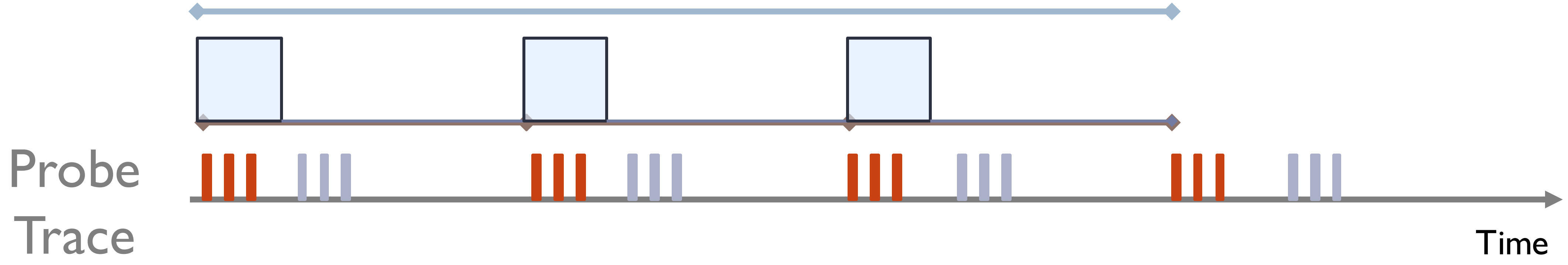
Create a template (base) pattern of where the probes will be



This *window size* is determined by the number of probes and their probe interval within a burst

Exploiting Burst Interval Patterns

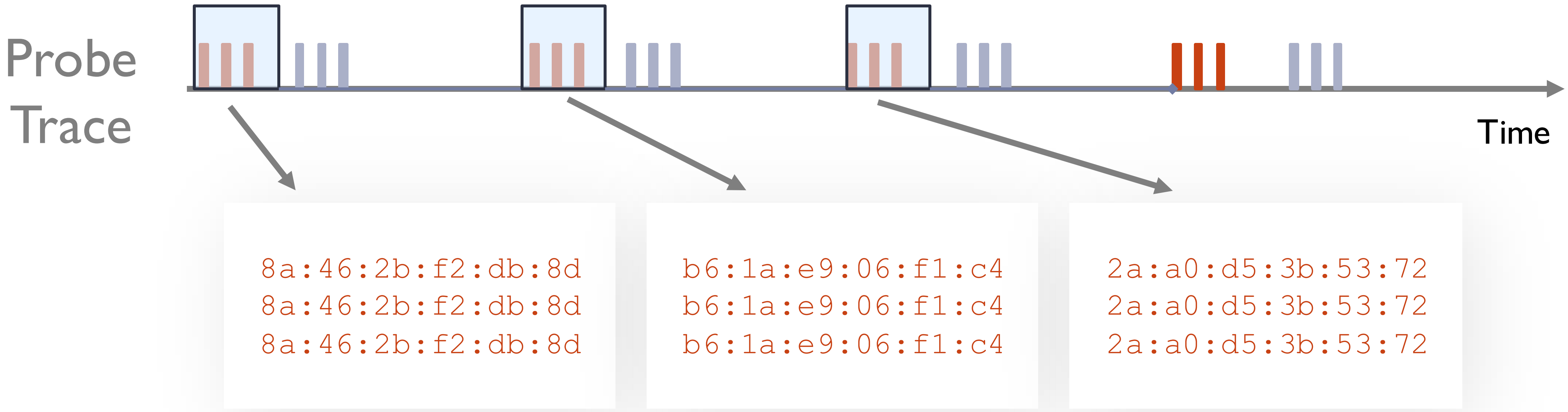
Create a template (base) pattern of where the probes will be



This *pattern length* is time in minutes to search a pattern for

Exploiting Burst Interval Patterns

Output the MAC addresses of probes matching this pattern

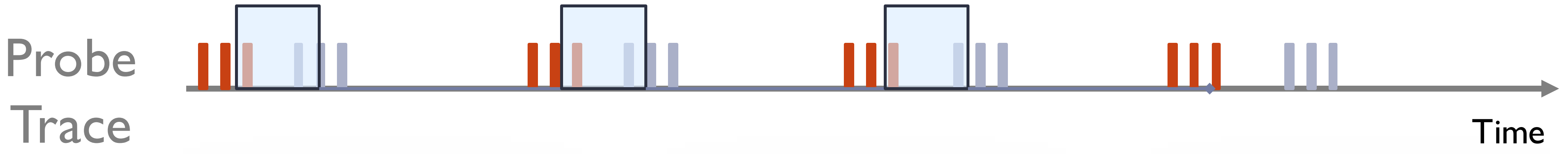


Exploiting Burst Interval Patterns

Time-shifted copies find the best match by amount of probes present

Exploiting Burst Interval Patterns

Time-shifted copies find the best match by amount of probes present



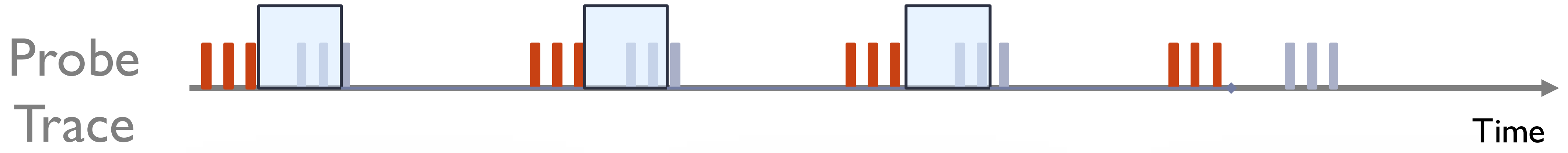
8a:46:2b:f2:db:8d
16:67:c1:04:39:bf

b6:1a:e9:06:f1:c4
54:8a:53:be:1d:df

2a:a0:d5:3b:53:72
9c:99:c4:cb:84:ea

Exploiting Burst Interval Patterns

Time-shifted copies find the best match by amount of probes present



16:67:c1:04:39:bf
16:67:c1:04:39:bf

54:8a:53:be:1d:df
54:8a:53:be:1d:df

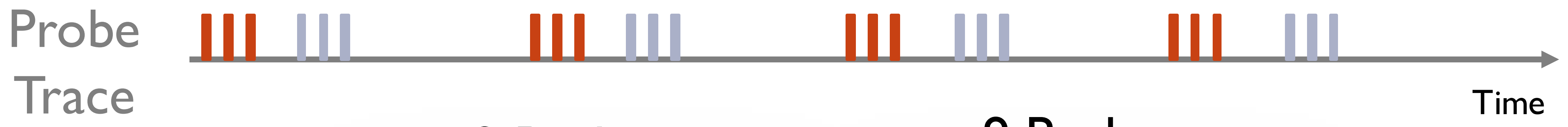
9c:99:c4:cb:84:ea
9c:99:c4:cb:84:ea

Exploiting Burst Interval Patterns

Time-shifted copies find the best match by amount of probes present

Exploiting Burst Interval Patterns

Find the **best** match by **# of probes**



9 Probes

```
8a:46:2b:f2:db:8d (3x)
b6:1a:e9:06:f1:c4 (3x)
2a:a0:d5:3b:53:72 (3x)
```

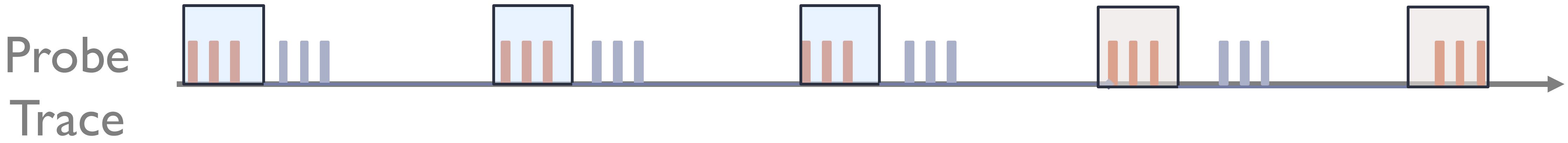
9 Probes

```
16:67:c1:04:39:bf (3x)
54:8a:53:be:1d:df (3x)
9c:99:c4:cb:84:ea (3x)
```

Highest ranking sets

Exploiting Burst Interval Patterns

Problem:
The pattern only finds probes within the length of the base pattern



Exploiting Burst Interval Patterns

Solution:

To extract longer sets, *iteratively chain* through them starting from the largest set to find probes belonging to the same device

8a:46:2b:f2:db:8d b6:1a:e9:06:f1:c4 2a:a0:d5:3b:53:72	16:67:c1:04:39:bf 54:8a:53:be:1d:df 9c:99:c4:cb:84:ea	a2:0a:5d:b3:35:27 8a:46:2b:f2:db:8d 6b:a1:9e:60:1f:50	16:67:c1:04:39:bf 6a:54:9f:23:41:0a 92:da:de:94:81:81

Exploiting Burst Interval Patterns

Solution:

To extract longer sets, *iteratively chain* through them starting from the largest set to find probes belonging to the same device

a2:0a:5d:b3:35:27	54:8a:53:be:1d:df	**8a:46:2b:f2:db:8d**	**16:67:c1:04:39:bf**
8a:46:2b:f2:db:8d	**16:67:c1:04:39:bf**	**6b:a1:9e:60:1f:50**	**9c:99:c4:cb:84:ea**
6b:a1:9e:60:1f:50	**9c:99:c4:cb:84:ea**	2a:a0:d5:3b:53:72	92:da:de:94:81:81

If there are intersecting MAC addresses, take the union to form a chain

Exploiting Burst Interval Patterns

Solution:

To extract longer sets, *iteratively chain* through them starting from the largest set to find probes belonging to the same device

a2:0a:5d:b3:35:27	54:8a:53:be:1d:df
8a:46:2b:f2:db:8d	**16:67:c1:04:39:bf**
6b:a1:9e:60:1f:50	**9c:99:c4:cb:84:ea**
2a:a0:d5:3b:53:72	92:da:de:94:81:81

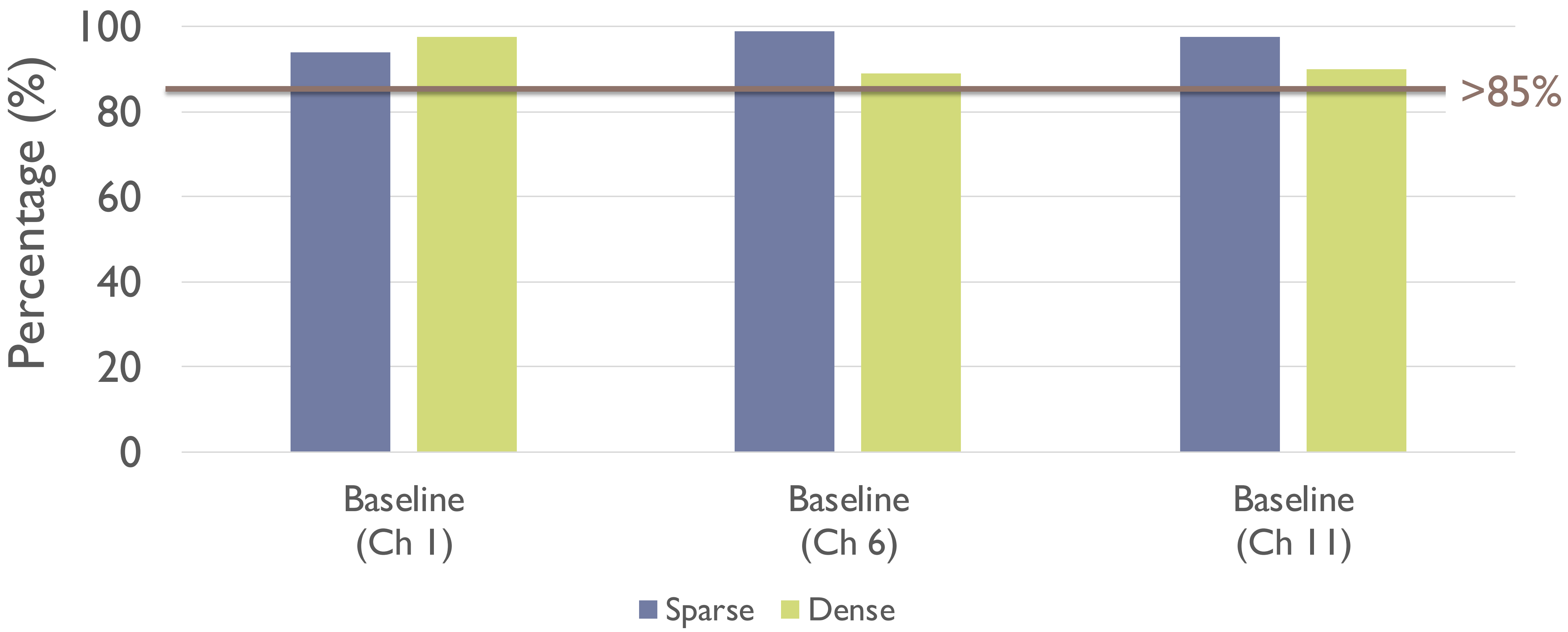
Result: Sets containing common probes across the packet trace

Metrics for Evaluation

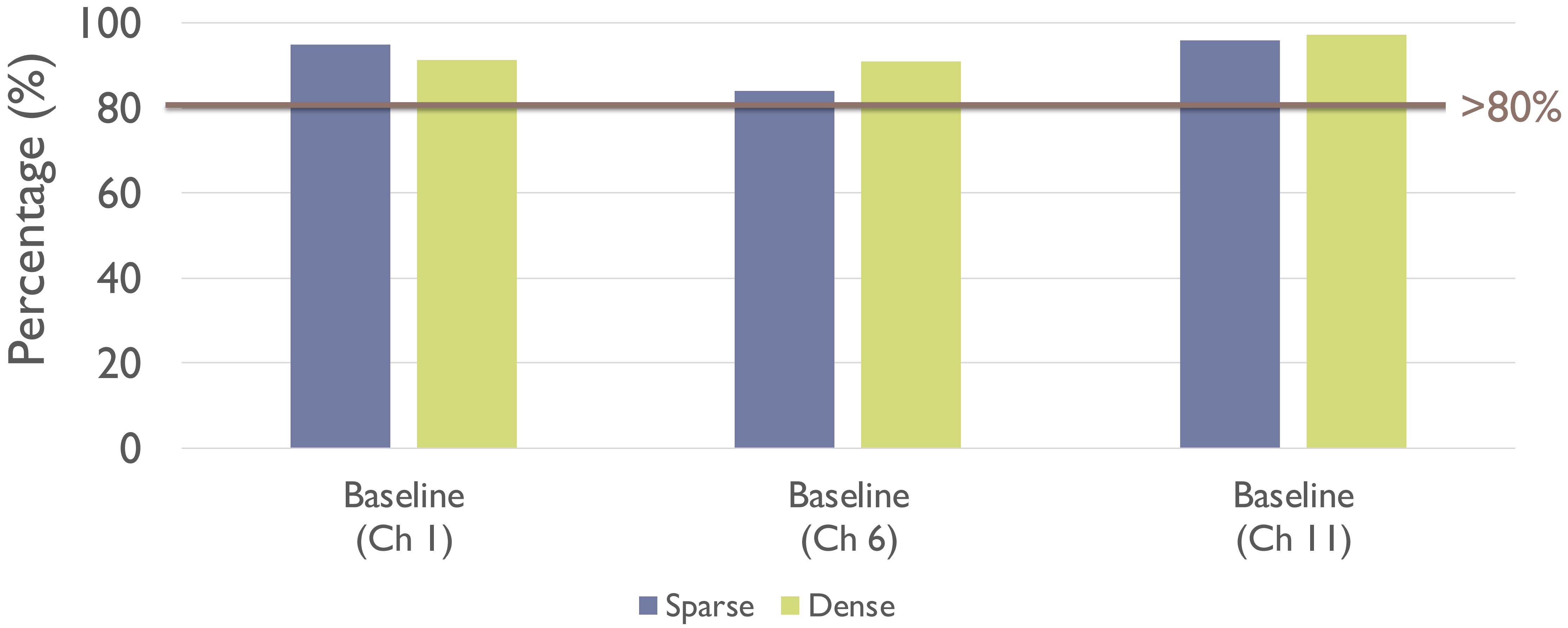
$$\textbf{Precision} \qquad \frac{\text{Correct matches}}{\text{Number of probes identified}}$$

$$\textbf{Recall} \qquad \frac{\text{Correct matches}}{\text{Total number of probes from the device in the trace}}$$

Precision – Burst Interval Attack



Recall – Burst Interval Attack



Example: Finding a Phone

Packet trace from
Pixel 7 Pro
160sec Burst Interval

Burst Interval Attack

Top set of MAC addresses

```
66:83:7f:77:a2:79 2
be:be:c2:5a:a5:69 2
52:5c:71:fc:35:71 2
52:ae:d3:4f:e6:10 2
ee:07:80:10:dc:2b 2
d2:97:06:0f:b5:dc 2
0a:0e:f5:a3:7b:d5 2
d2:f3:45:d4:a6:84 2
5e:8d:68:82:02:5e 2
```

18/20 identified
2 missed from the end
from timing drift

Example: Finding a Phone

Packet trace from
Pixel 7 Pro
160sec Burst Interval

Burst Interval Attack

Top set of MAC addresses

```
66:83:7f:77:a2:79 2
be:be:c2:5a:a5:69 2
52:5c:71:fc:35:71 2
52:ae:d3:4f:e6:10 2
ee:07:80:10:dc:2b 2
d2:97:06:0f:b5:dc 2
0a:0e:f5:a3:7b:d5 2
d2:f3:45:d4:a6:84 2
5e:8d:68:82:02:5e 2
```

Timing attacks are effective even with MAC randomization

nd
from timing drift

Jittery: a set of Wi-Fi privacy defense mechanisms

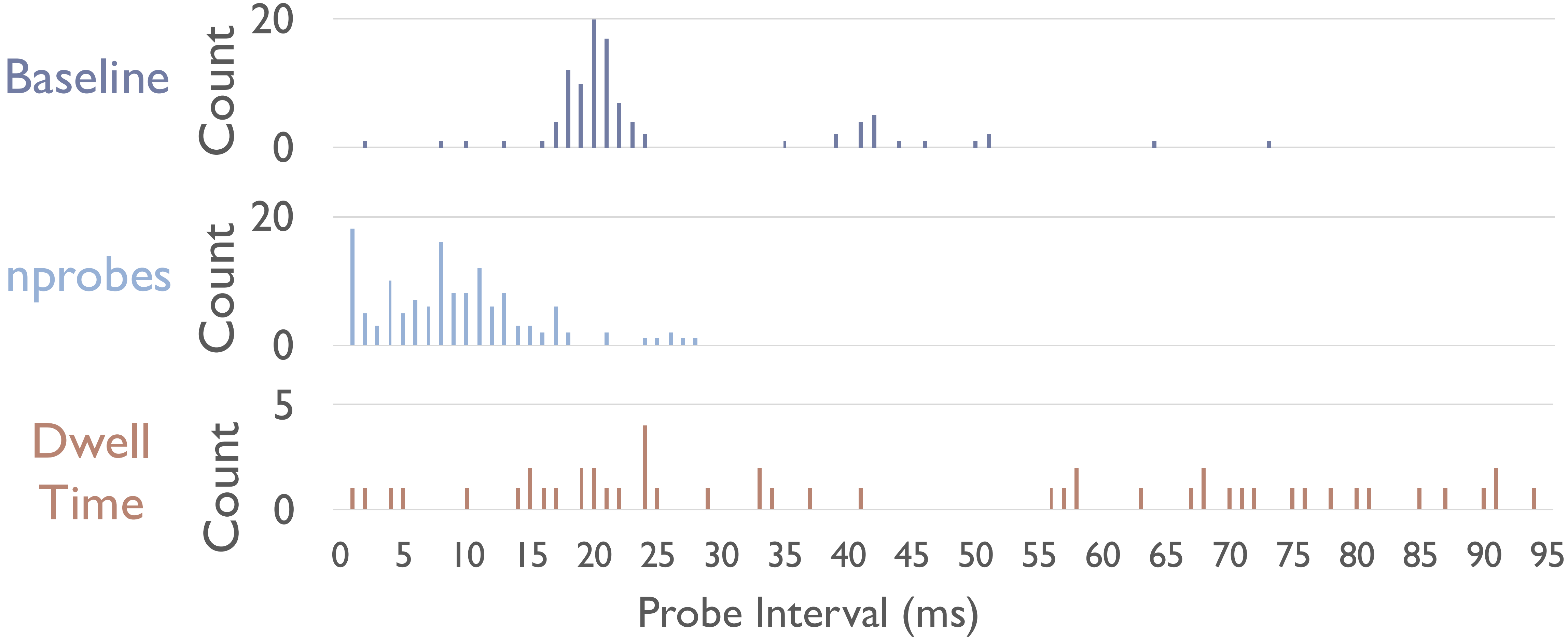
- Recovers MAC randomization privacy benefits
  - Break timing patterns in network discovery
- Randomize built-in parameters of 802.11
  - MAC Randomization on all 6 bytes of the source address
  - Number of probes per burst (nprobes)
  - Random dwell time (1-100ms)
  - Shuffled channel ordering
  - Dynamic burst intervals
- No changes to infrastructure
- Potential for standardization in MAC randomization

Driver-level Implementation

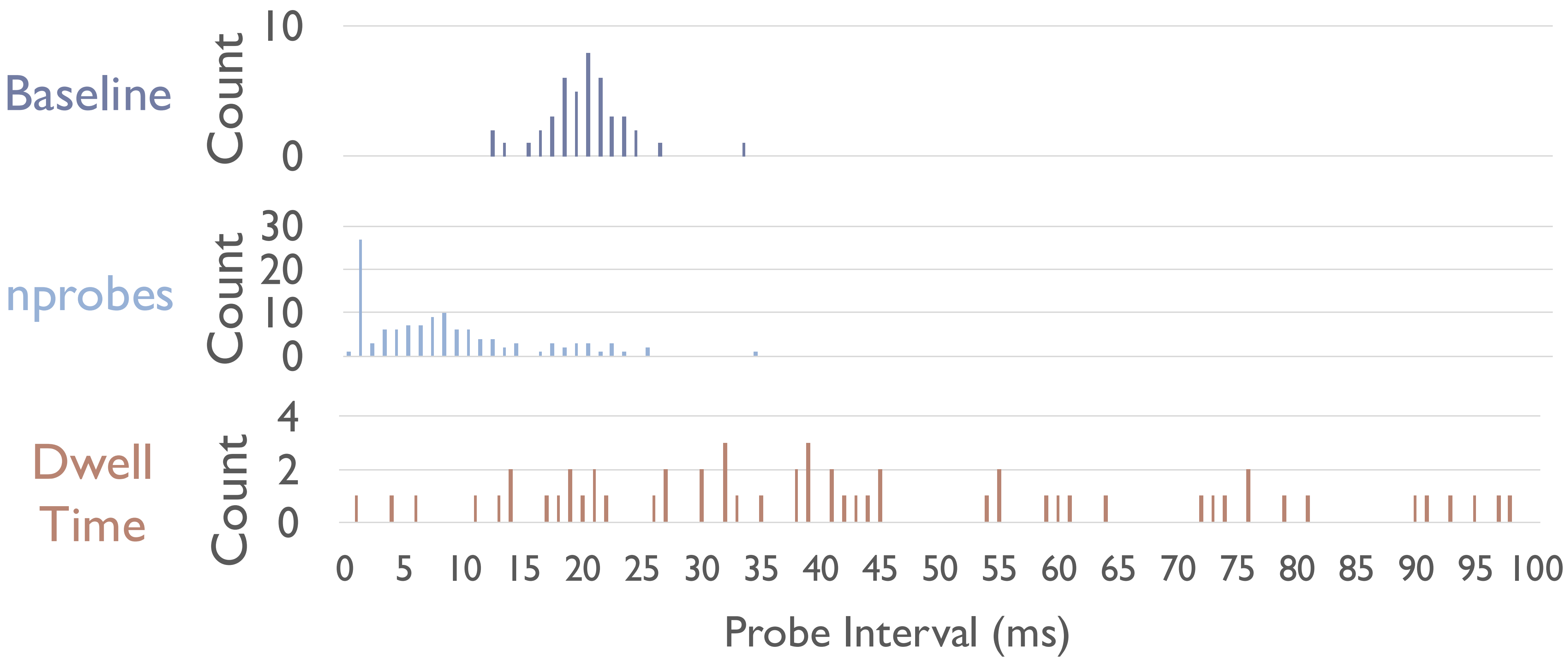
- Modified *brcmfmac* driver deployed on Raspberry Pi 3B+ devices

- Burst interval modifications tested with Netlink

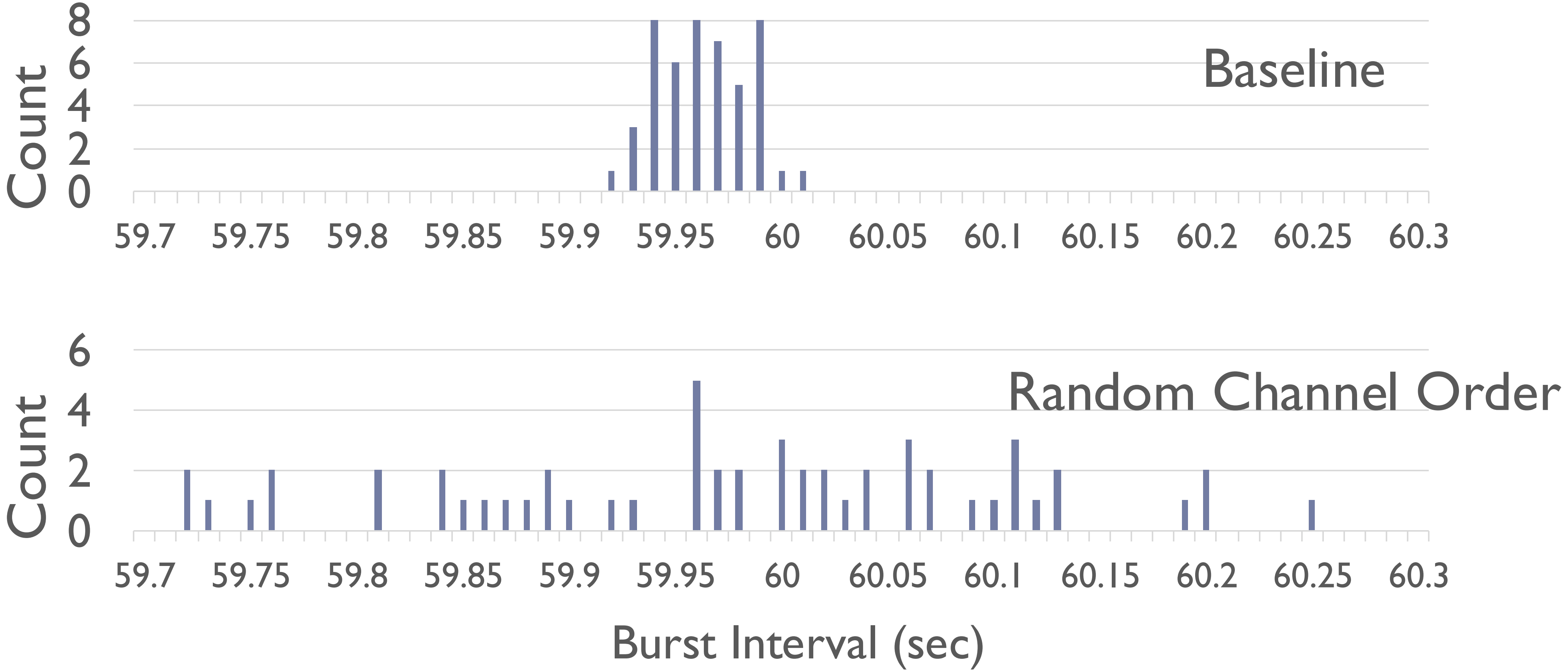
Probe Interval Distribution: Sparse



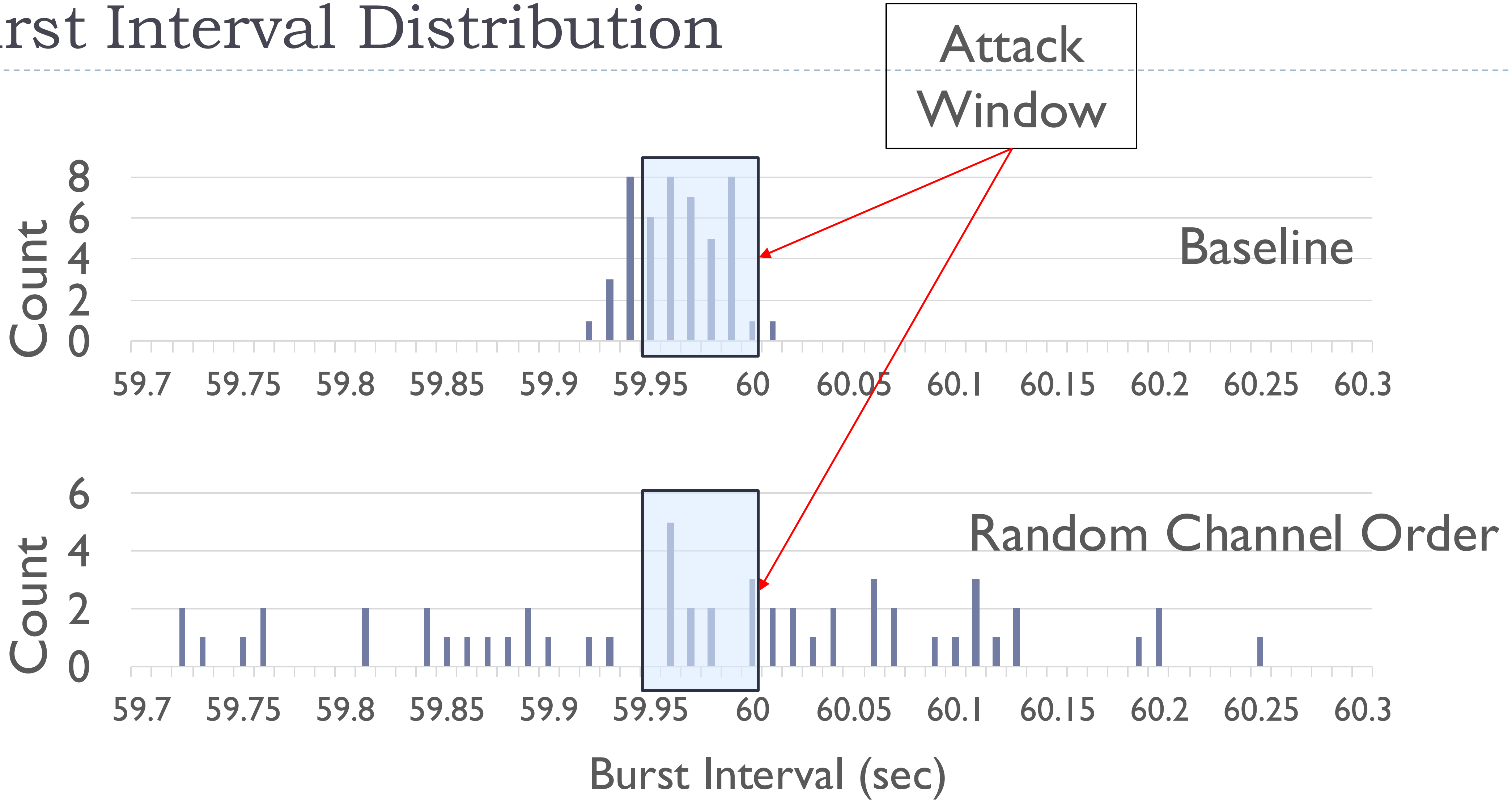
Probe Interval Distribution: Dense



Burst Interval Distribution



Burst Interval Distribution



Metrics for Evaluation

Precision

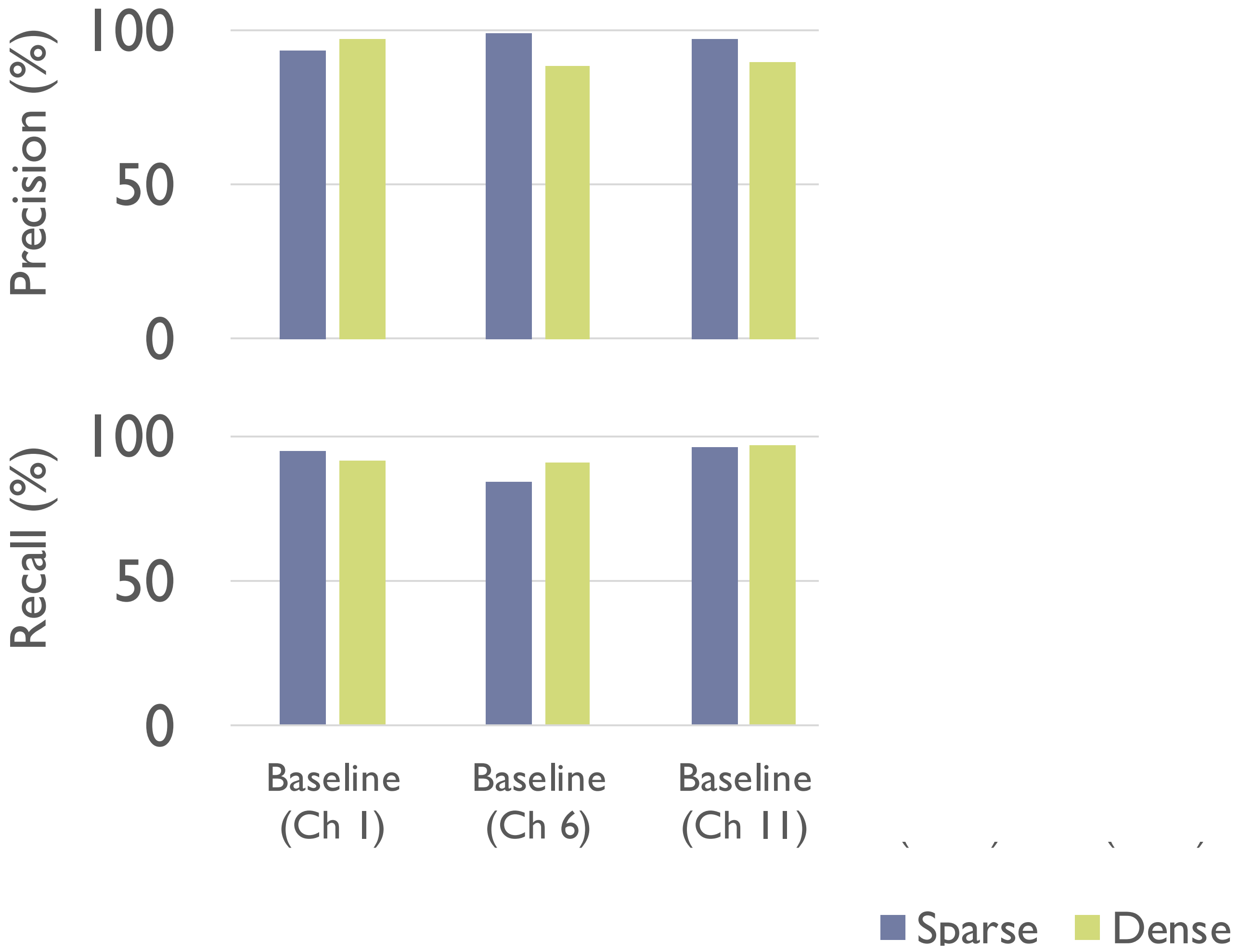
$$\frac{\text{Correct matches in probes identified}}{\text{Number of probes identified}}$$

Recall

$$\frac{\text{Correct matches in probes identified}}{\text{Total number of probes from our device in the trace}}$$

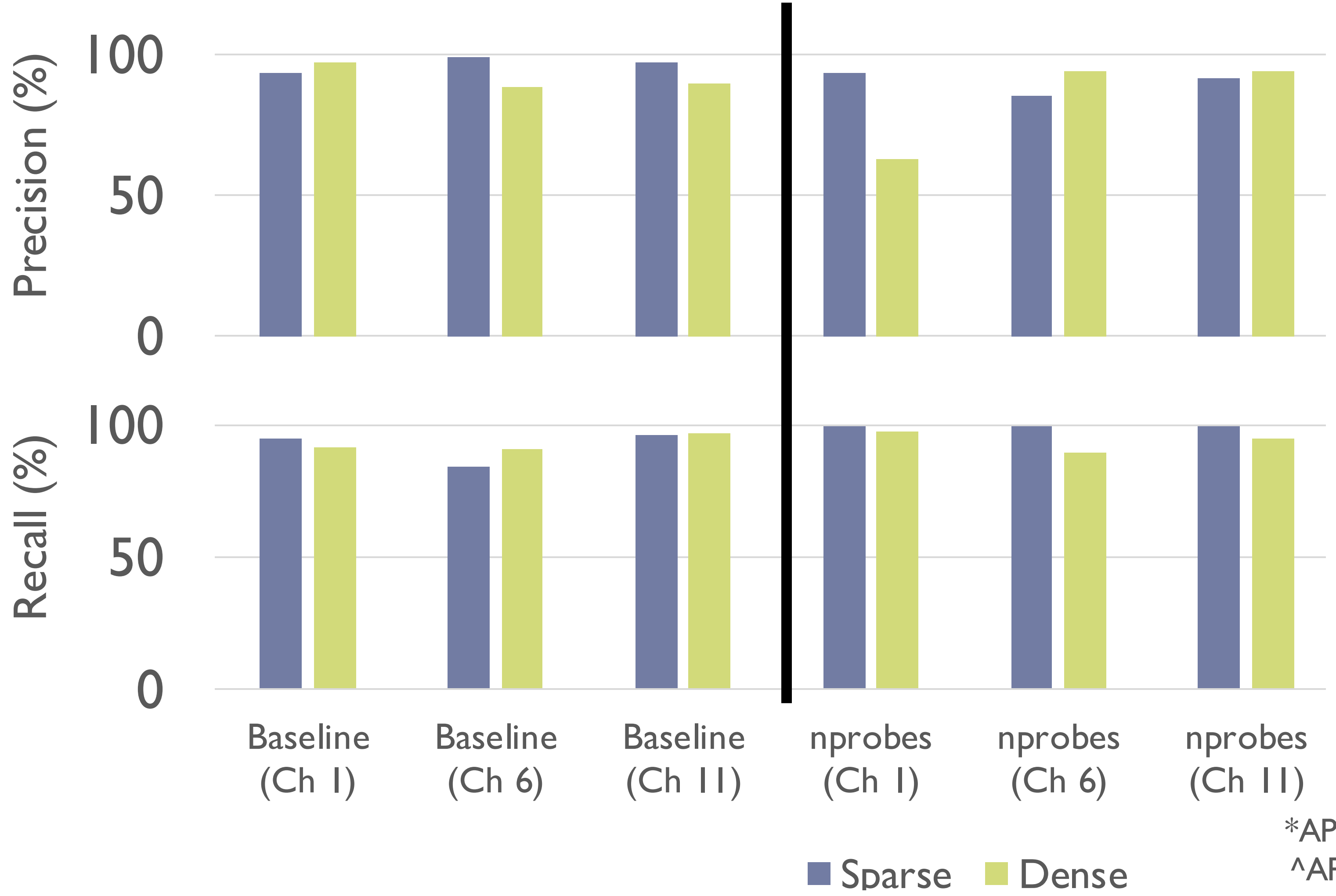
AP Discovery Rate	% of APs responsive to a probe event in 100ms
Energy consumption	Average number of probes per burst
AP Discovery Delay	Average and Median time to receive a probe response

Burst Interval Attack Results



*APs were not operating in the sparse environment
^APs were not operating in the dense environment

Burst Interval Attack Results

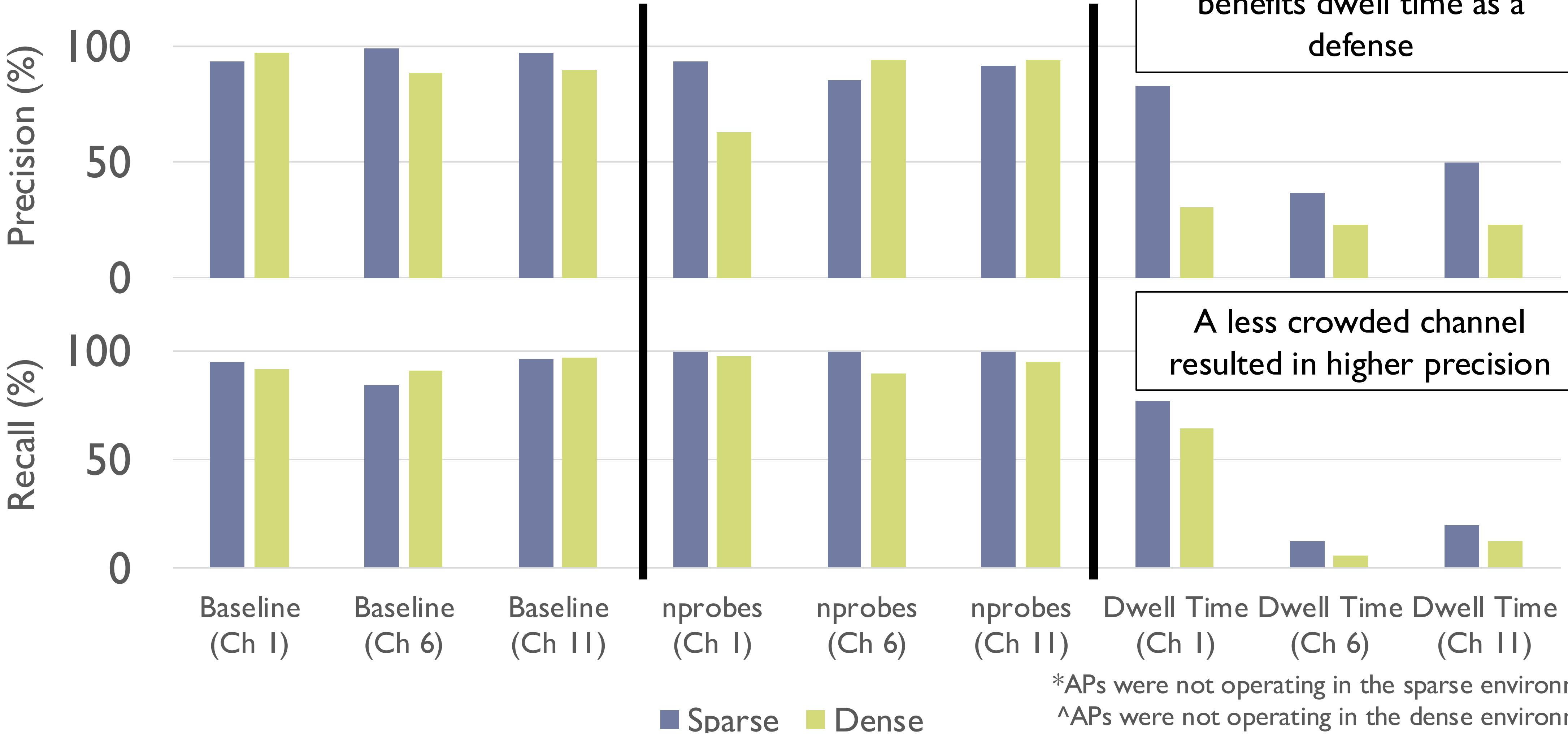


More probes on Channel 1 results in a higher FPR

Random nprobes is not enough to be hidden

*APs were not operating in the sparse environment
^APs were not operating in the dense environment

Burst Interval Attack Results



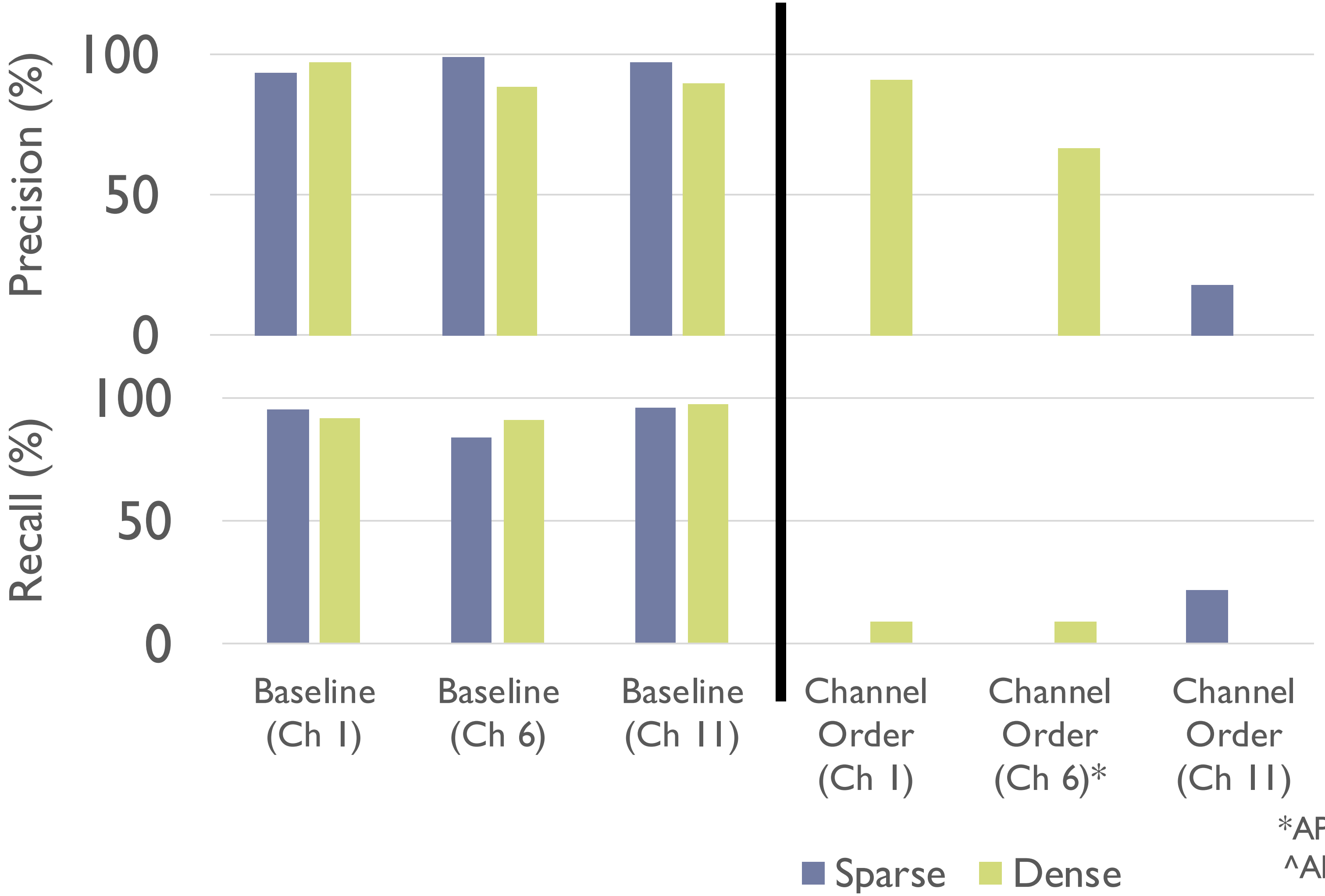
A more crowded channel benefits dwell time as a defense

A less crowded channel resulted in higher precision

*APs were not operating in the sparse environment

^APs were not operating in the dense environment

Burst Interval Attack Results



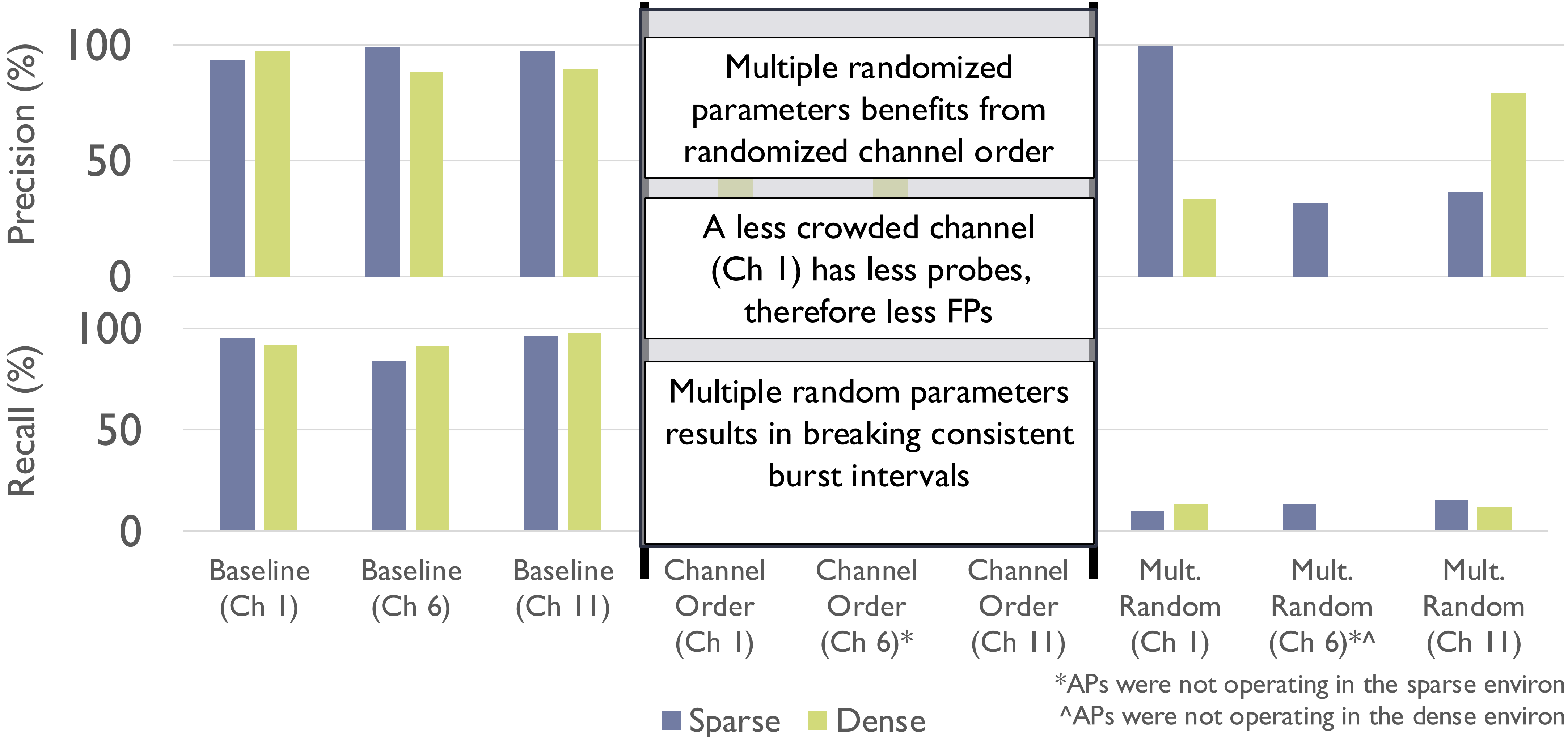
Random channel order breaks the burst interval pattern

Additional contention increases burst interval randomness

Random channel order greatly benefits sparse networks

*APs were not operating in the sparse environment
^APs were not operating in the dense environment

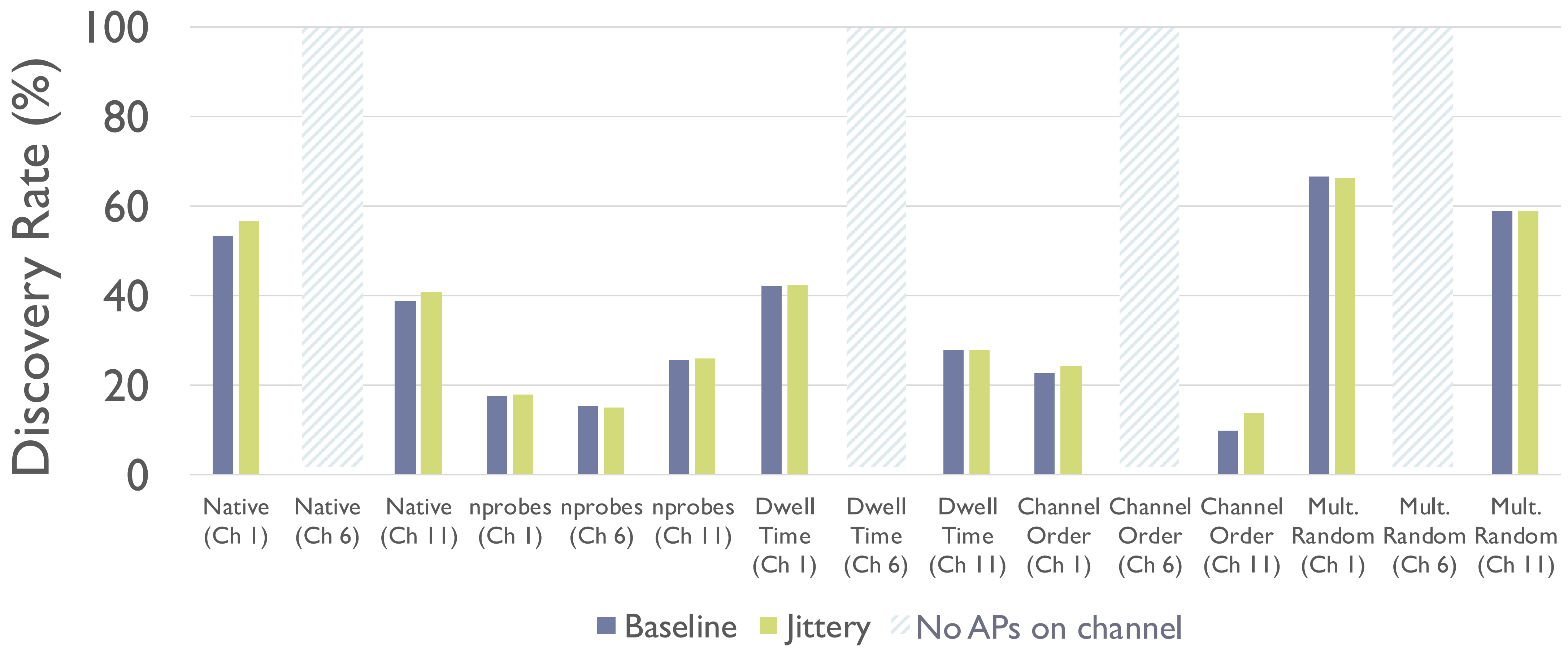
Burst Interval Attack Results



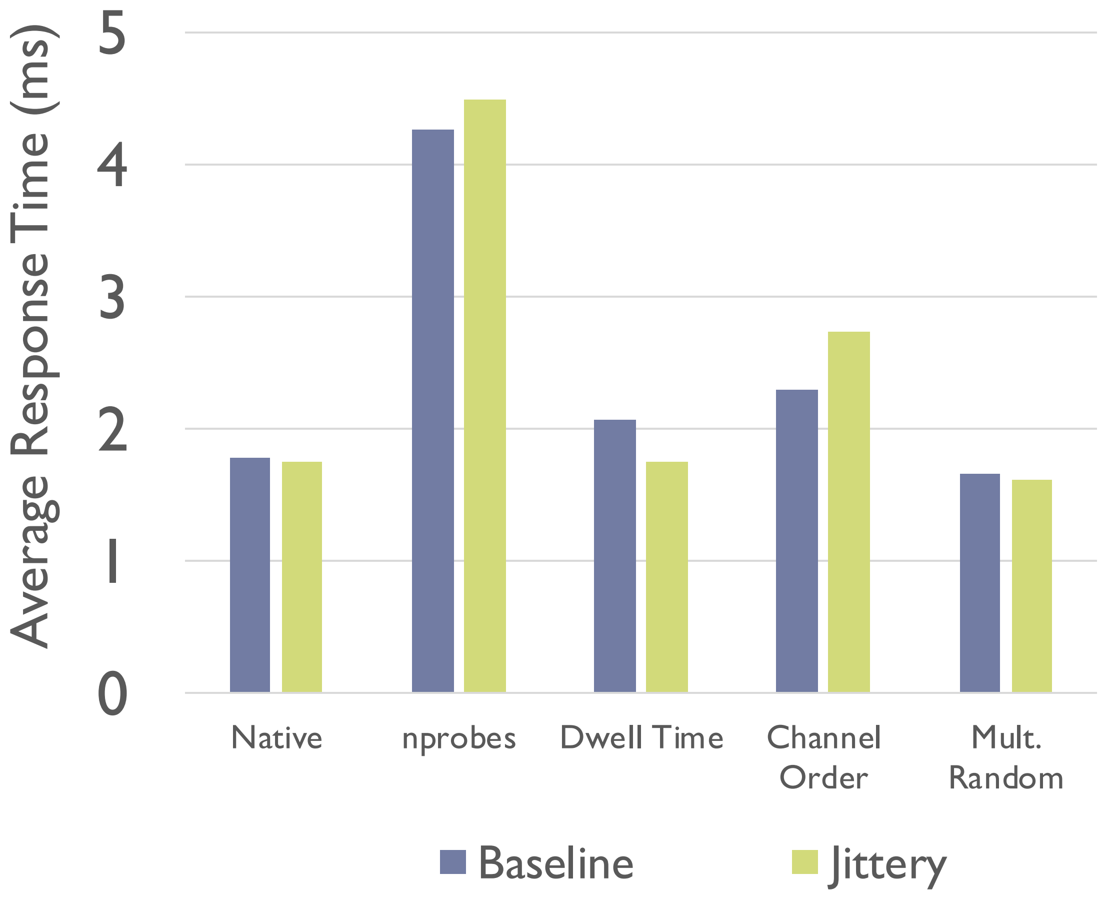
*APs were not operating in the sparse environment

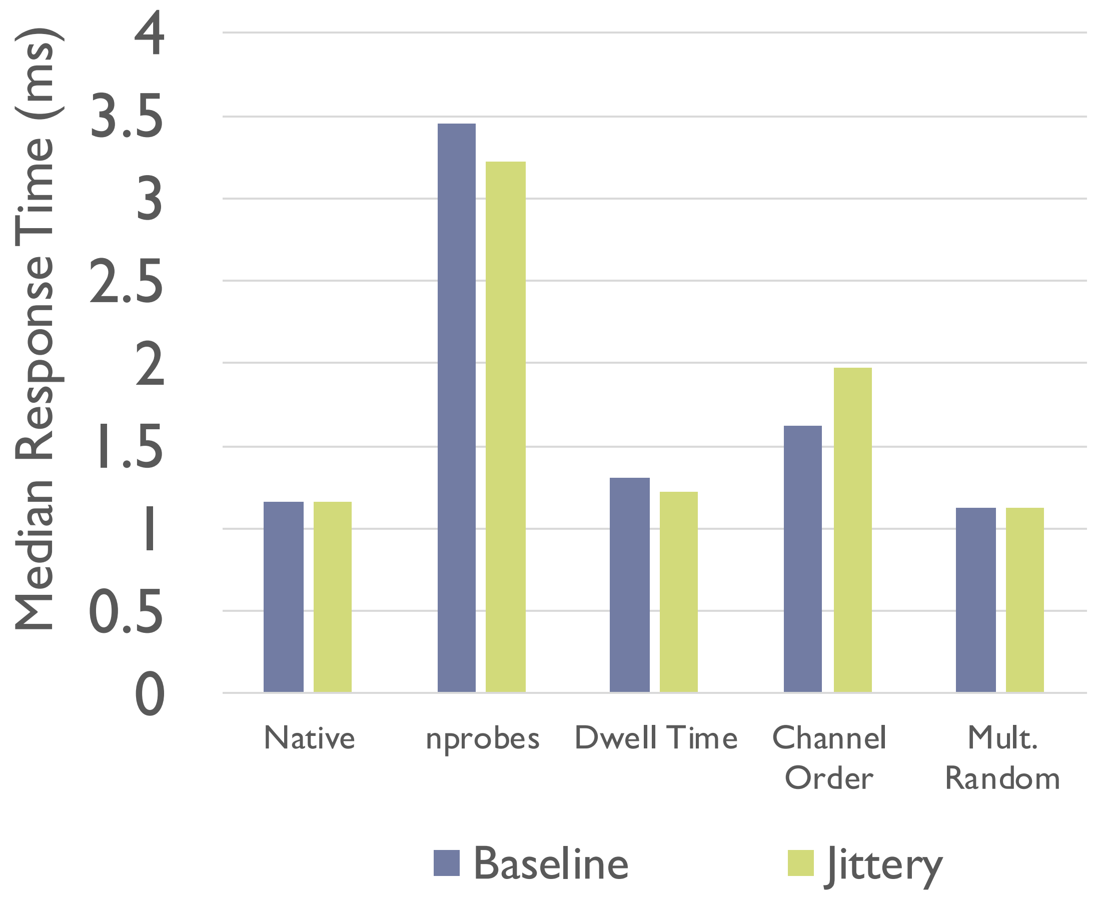
^APs were not operating in the dense environment

AP Discovery Rates - Sparse

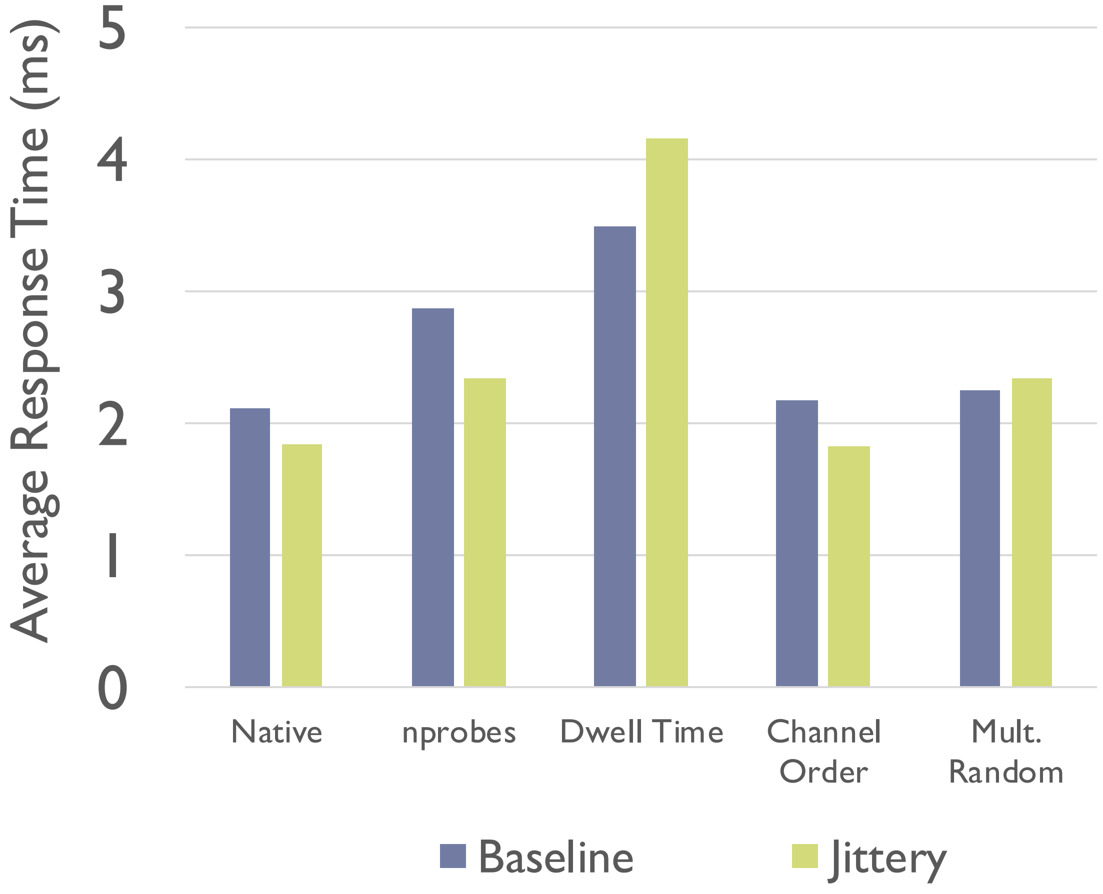


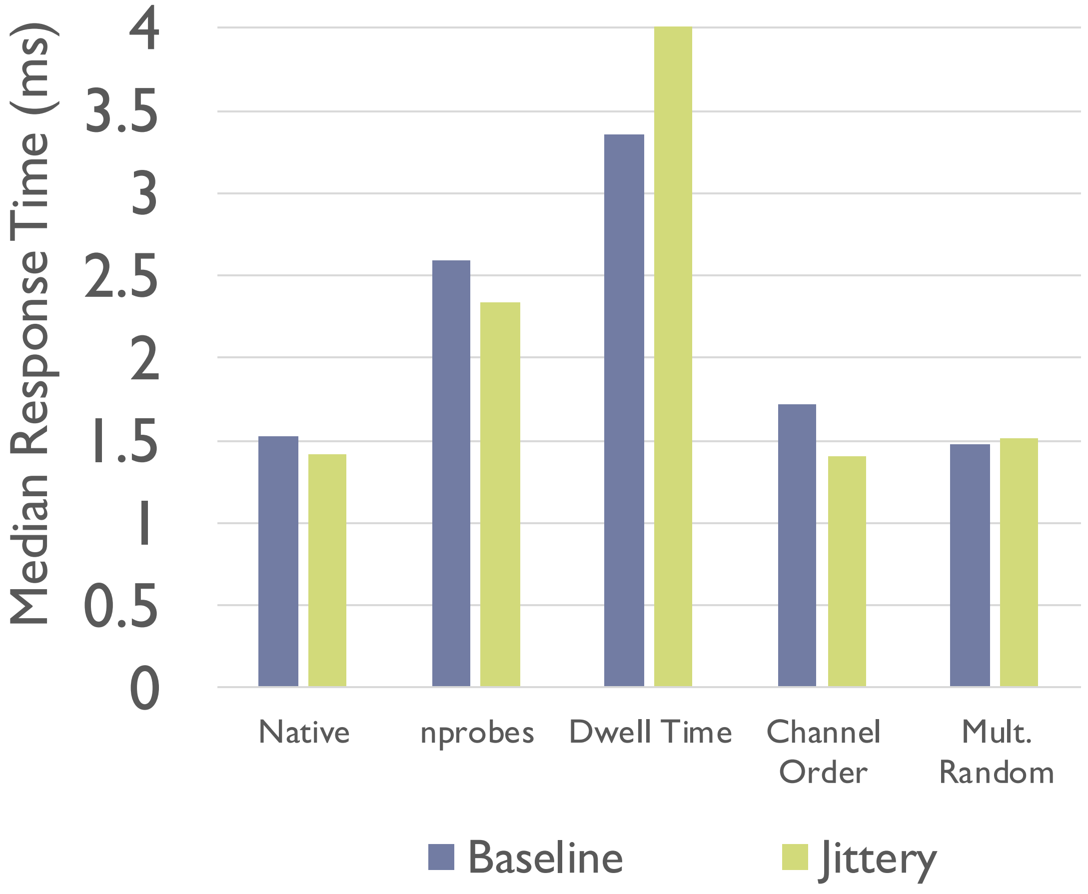
AP Discovery Delay – Sparse Networks



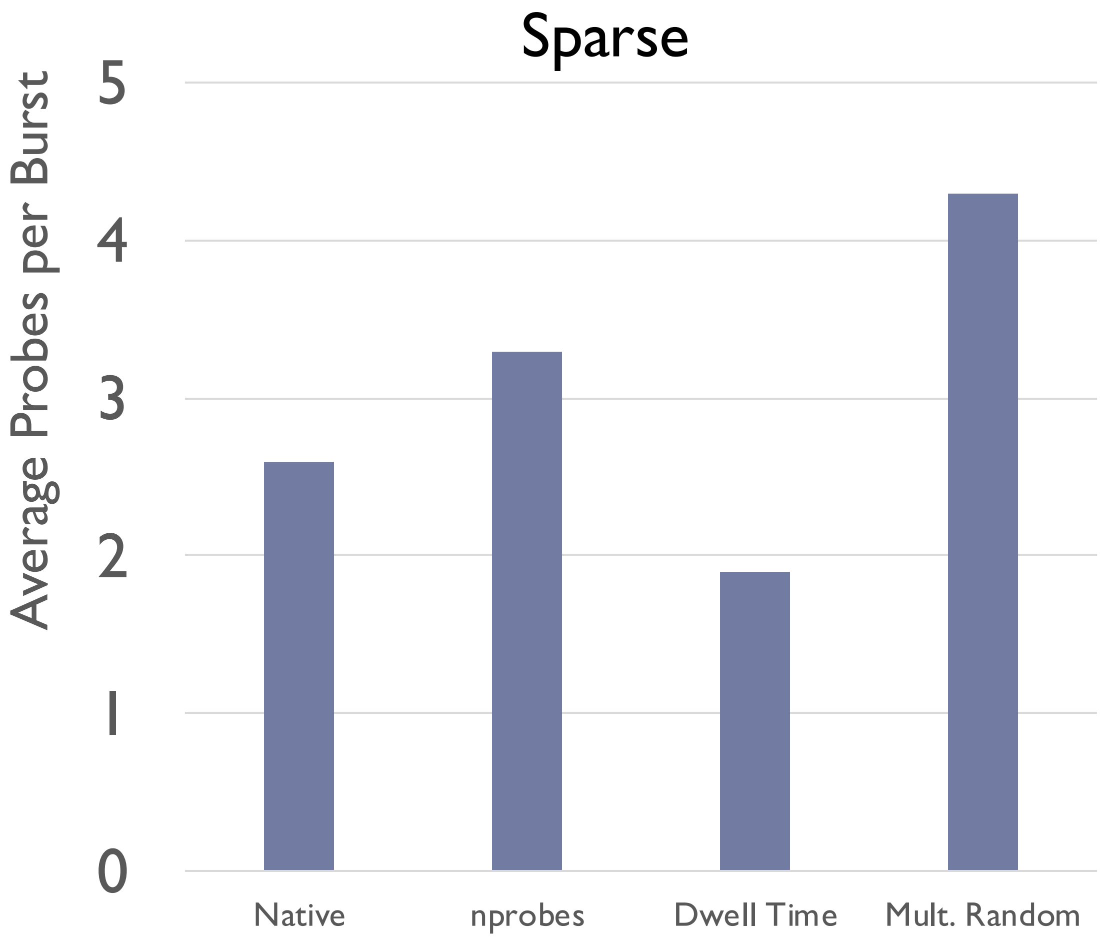


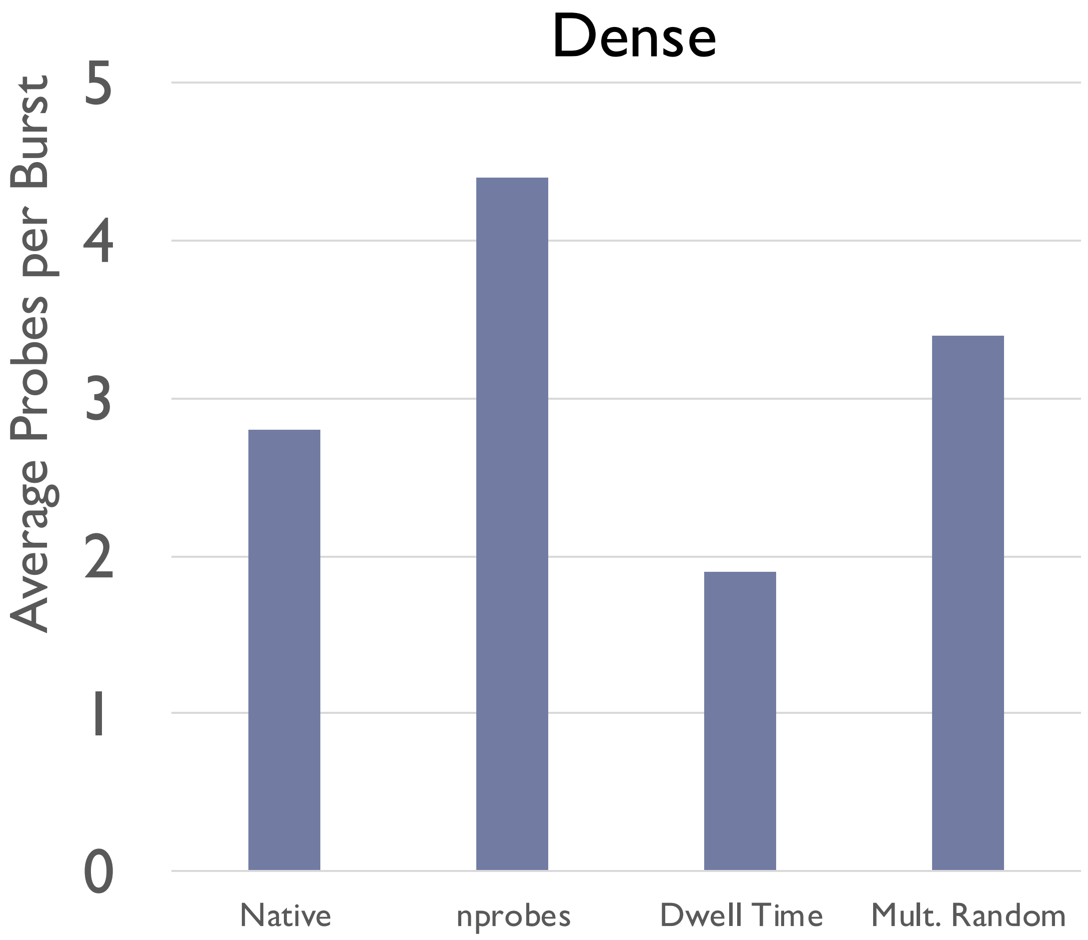
AP Discovery Delay – Dense Networks



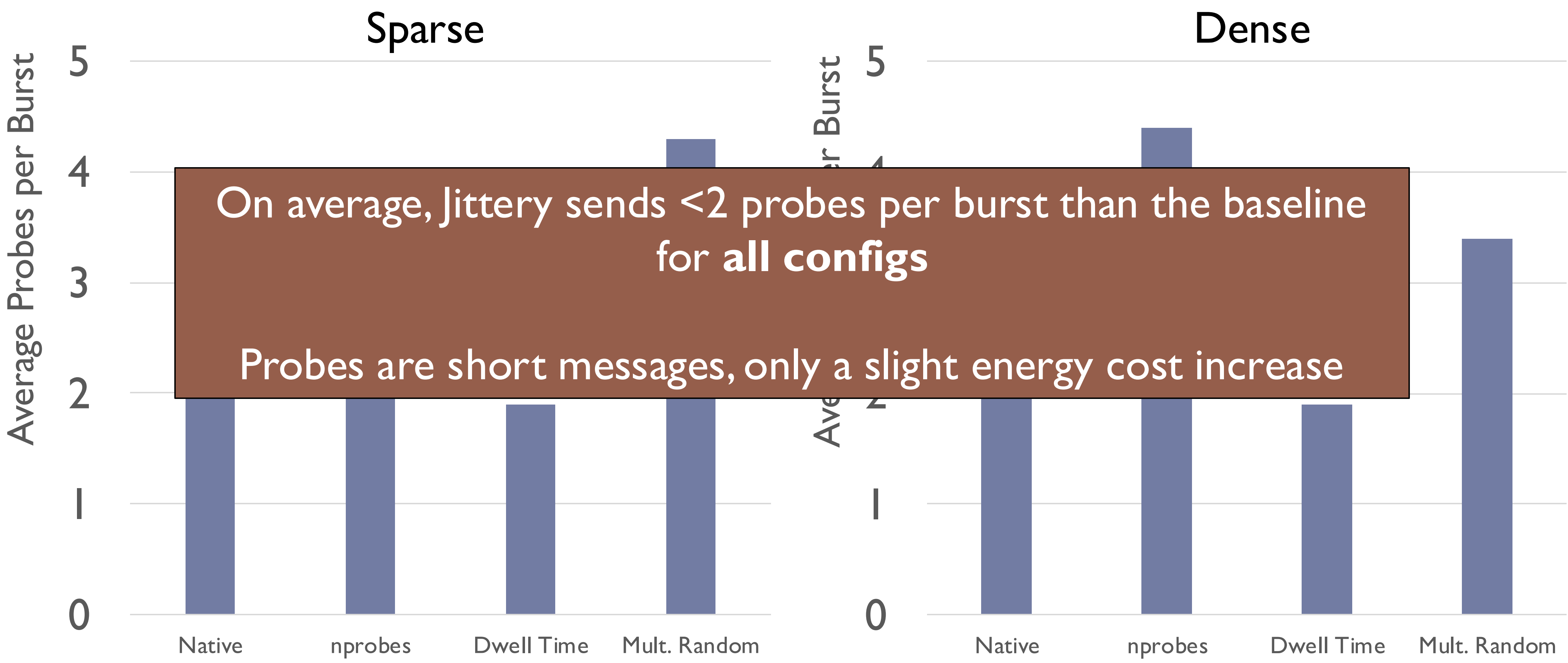


Energy Consumption





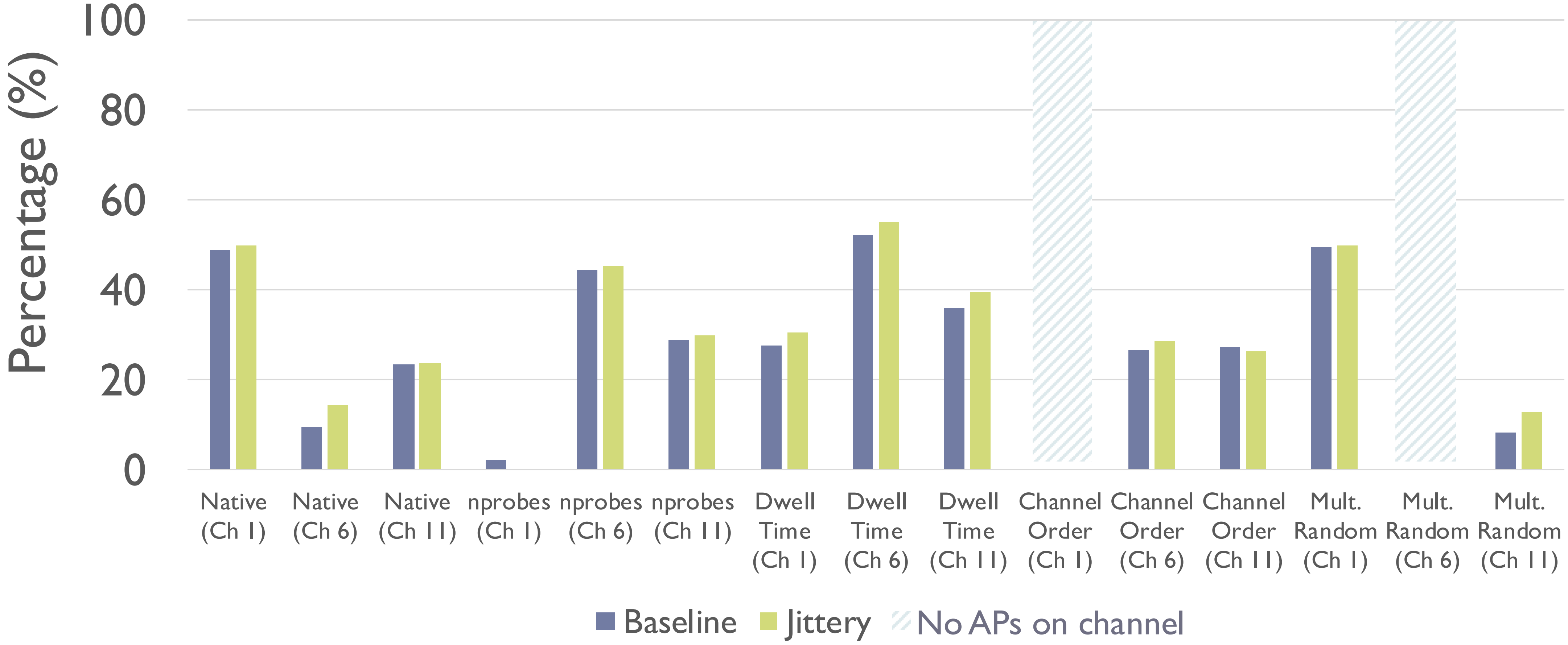
Energy Consumption



On average, Jittery sends <2 probes per burst than the baseline for **all configs**

Probes are short messages, only a slight energy cost increase

AP Discovery Rates - Dense



Future Directions

- Identifying devices from the same vendor with higher accuracy may require additional metrics that are not timing-based
  - We constrain the attack to solely use timing metrics
  - Future approaches may expand this by using other data fields *with* timing
- Signal strength of the probe responses is a factor in calculating successful AP discovery rate
  - Reported results could be lower than actual due to monitor devices not receiving probe responses

Recommendations

- Configure network discovery with
  - Random sequence numbers
  - Changing number of probes each burst
  - Variable dwell time per burst
  - Variable burst intervals
- Randomize the full length of the MAC address (48 bits)
- Change the MAC address each burst in network discovery
- Eliminate using directed probes
- Offload features from IEs to the Association phase