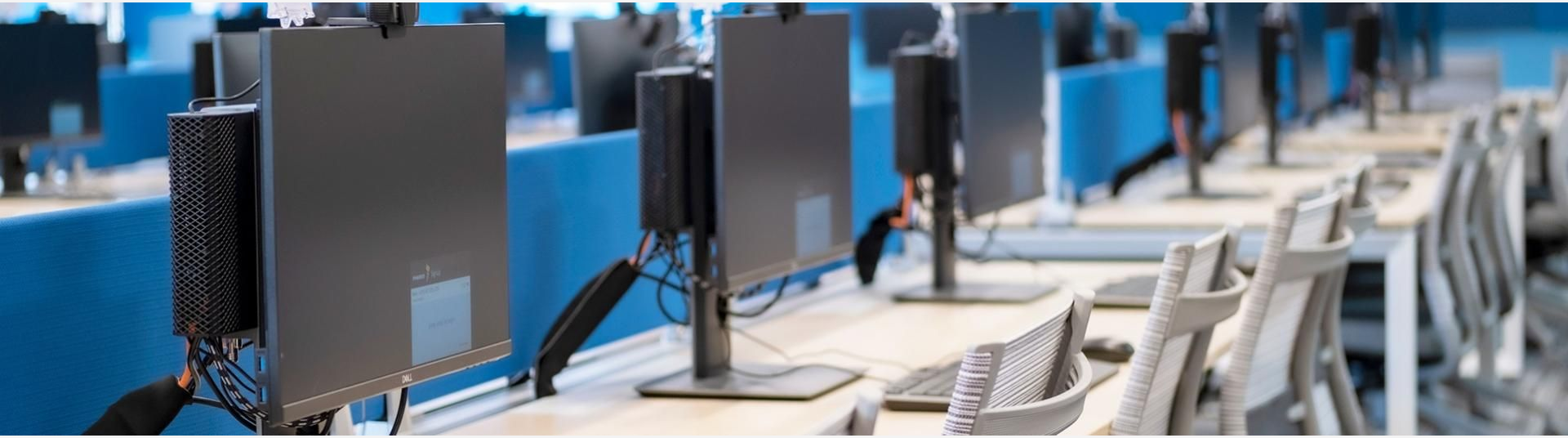
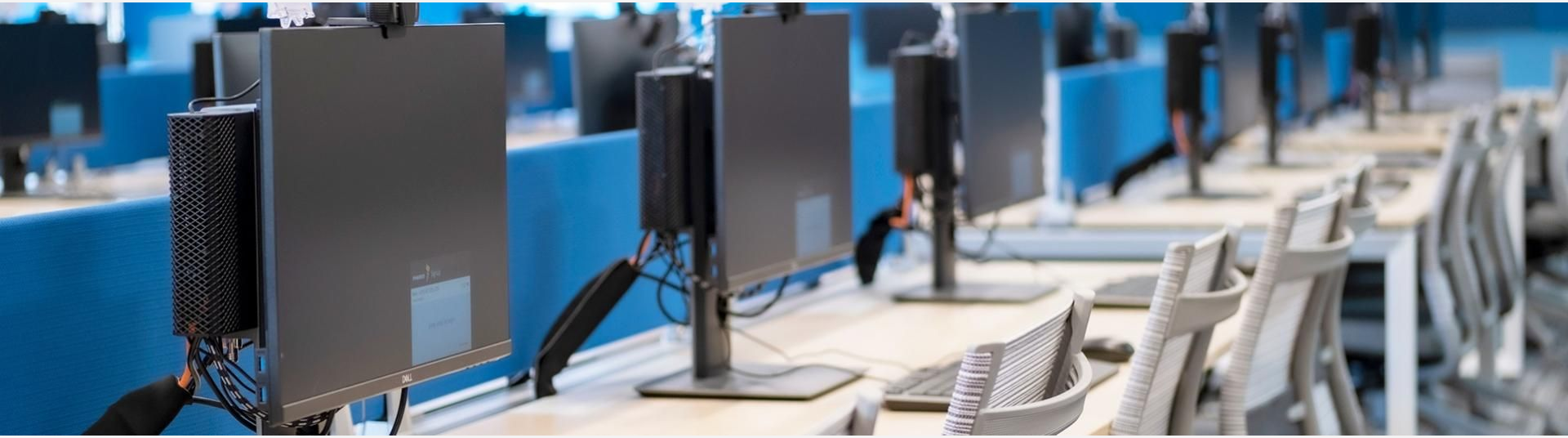
TagAlong: A Free, Wide-Area Data-Muling Service Built on the AirTag Protocol

Alex Bellon, *Alex Yen*, *Pat Pannuto*

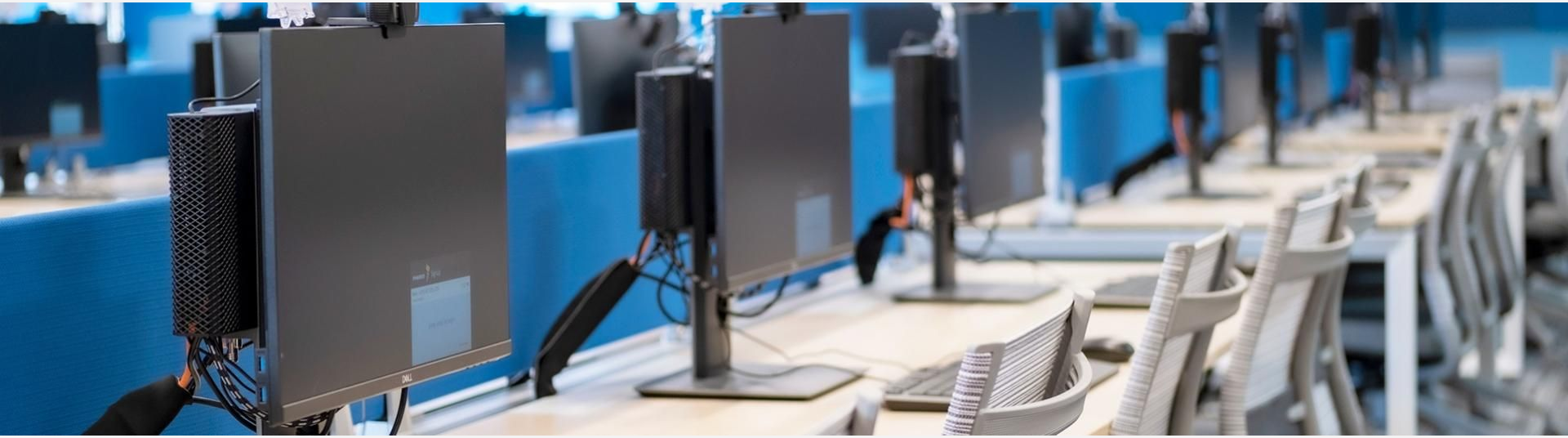
UC San Diego



- Track real time occupancy, without setting up backhaul infrastructure



- Track real time occupancy, without setting up backhaul infrastructure
- Want to use data muling - device to carry the from sensors



- Track real time occupancy, without setting up backhaul infrastructure
- Want to use data muling - device to carry the from sensors
- We’re going to enable that for free, with existing infrastructure


Bluetooth LE • Ultra Wideband • AirTag • Designed by Apple in California • Assembled in China • Bluetooth LE • Ultra Wideband • AirTag • Designed by Apple in California • Assembled in China •

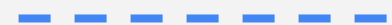

MacBook Pro

$(priv_0, pub_0)$ on P-224
+
symmetric key
=
master beacon key

[0] A. Heinrich, M. Stute, T. Kornhuber, and M. Hollick. Who Can Find My Devices? Security and Privacy of Apple's Crowd-Sourced Bluetooth Location Tracking System. Proceedings on Privacy Enhancing Technologies, 2021(3):227–245, 2021

0xabcdabcd...cd

28 byte pub_0



0xcafef00d...0d

28 byte pub_1

00:15

0xdeadbeef...00

28 byte pub_2

00:30


0xdeadbeef...00
28 byte pub_2


0xdeadbeef…00
28 byte pub_2
location report
encrypted with derived key

{0xdeadbeef…00: enc_location_report_0}

{0xdeadbeef…00: enc_location_report_0}
{0xcafef00d…01: enc_location_report_1}
{0x01234567…02: enc_location_report_2}
{0xfaceface…03: enc_location_report_3}
{0xdeadbeef…04: enc_location_report_4}
{0xdeadbeef…05: enc_location_report_5}
{0xdeadbeef…06: enc_location_report_6}

{0xdeadbeef…00: enc_location_report_0}
{0xcafef00d…01: enc_location_report_1}
{0x01234567…02: enc_location_report_2}
{0xfaceface…03: enc_location_report_3}
{0xdeadbeef…04: enc_location_report_4}
{0xdeadbeef…05: enc_location_report_5}
{0xdeadbeef…06: enc_location_report_6}

{0xdeadbeef…00: enc_location_report_0}
{0xcafef00d…01: enc_location_report_1}
{0x01234567…02: enc_location_report_2}
{0xfaceface…03: enc_location_report_3}
{0xdeadbeef…04: enc_location_report_4}
{0xdeadbeef…05: enc_location_report_5}
{0xdeadbeef…06: enc_location_report_6}

{0xdeadbeef…00}

{0xdeadbeef…00: enc_location_report_0}
{0xcafef00d…01: enc_location_report_1}
{0x01234567…02: enc_location_report_2}
{0xfaceface…03: enc_location_report_3}
{0xdeadbeef…04: enc_location_report_4}
{0xdeadbeef…05: enc_location_report_5}
{0xdeadbeef…06: enc_location_report_6}

{0xdeadbeef…00}

{0xdeadbeef…00: enc_location_report_0}
{0xcafef00d…01: enc_location_report_1}
{0x01234567…02: enc_location_report_2}
{0xfaceface…03: enc_location_report_3}
{0xdeadbeef…04: enc_location_report_4}
{0xdeadbeef…05: enc_location_report_5}
{0xdeadbeef…06: enc_location_report_6}

{0xdeadbeef…00} {enc_location_report_0}

0x00000008...01
28 byte pub_i

location report
encrypted with derived key

```
00000008 00000002 deadbeef 0000000000000000000000000000 01
```

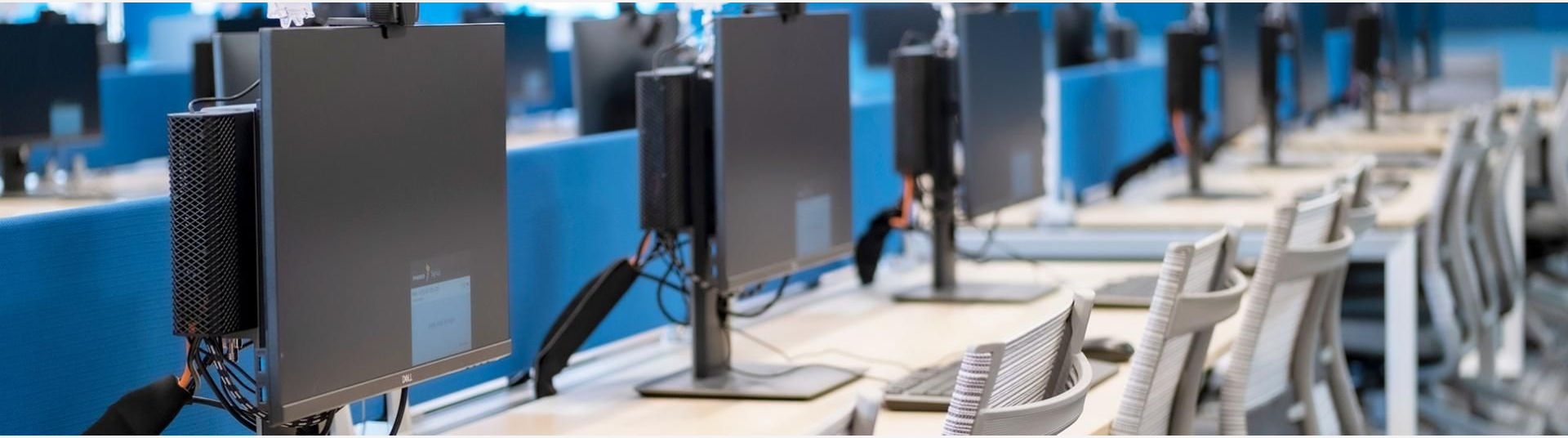
bit index | message ID | modem ID | padding 0s | bit

[1] F. Braeunlein. *Send My: Arbitrary data transmission via Apple's Find My network*, Positive Security

```
00000008 00000002 deadbeef 00000000 00000000000000000000 01
```

bit index | message ID | modem ID | counter | padding 0s | bit

- Only one bit of data per BLE advertisement packet
- Only one byte of payload is being used for actual data, lots of padding
- Bit index, message ID and modem ID all have an upper limit
  - 16^8 = 4,294,967,296 bits, messages or modems



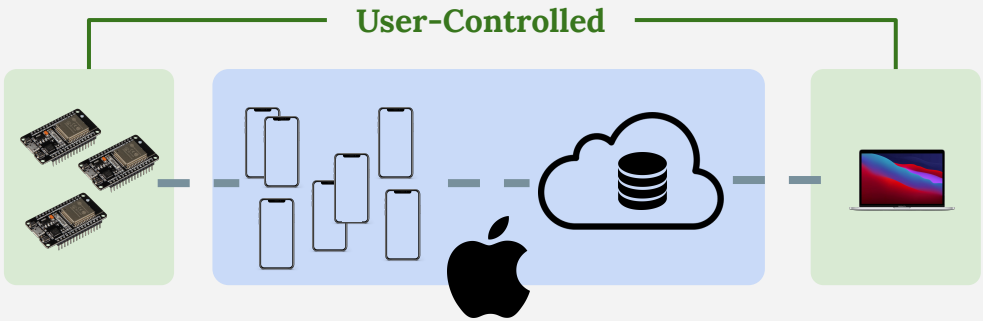
What primitives would we want in this situation?

- Unlimited number of messages/modems/data, increased data rate, greater payload sizes, etc.

TagAlong

Provide scalable arbitrary data transmission using Find My network

- Use existing Find My infra to remove limits on messages, modems, etc.
- Variable payload sizes to accommodate different situations
- More efficient usage of BLE advertisement packet



Use existing Find My infrastructure to remove limits on **messages, modems**, bit length

00:00	pub_0: 0xabcdabcd…cd
00:15	pub_1: 0xcafef00d…0d
00:30	pub_2: 0xdeadbeef…00
00:45	pub_3: 0xfaceface…ce
01:00	pub_4: 0x01234567…ef

Use existing Find My infrastructure to remove limits on **messages, modems**, bit length

00:00	pub_0:	0xabcdabcd…cd	= start of $message_0$
00:15	pub_1:	0xcafef00d…0d	= start of $message_1$
00:30	pub_2:	0xdeadbeef…00	= start of $message_2$
00:45	pub_3:	0xfaceface…ce	= start of $message_3$
01:00	pub_4:	0x01234567…ef	= start of $message_4$

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

0x0000000000000000000000000000000000000000000000000000000000000000

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x000000000000000000000000000000000000000000000000000000000000000ef
```

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x00000000000000000000000000000000000000000000000000000000000000ef
0x000000000000000000000000000000000000000000000000000000000000beef
```

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x00000000000000000000000000000000000000000000000000000000000000ef
0x000000000000000000000000000000000000000000000000000000000000beef
0x0000000000000000000000000000000000000000000000000000000000adbeef
```

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x00000000000000000000000000000000000000000000000000000000000000ef
0x000000000000000000000000000000000000000000000000000000000000beef
0x0000000000000000000000000000000000000000000000000000000000adbeef
0x00000000000000000000000000000000000000000000000000000000deadbeef
```

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x00000000000000000000000000000000000000000000000000000000000000ef
0x000000000000000000000000000000000000000000000000000000000000beef
0x0000000000000000000000000000000000000000000000000000000000adbeef
0x00000000000000000000000000000000000000000000000000000000deadbeef

                                    ...

0x0000beefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadbefe
```

Use existing Find My infrastructure to remove limits on messages, modems, **bit length**

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x00000000000000000000000000000000000000000000000000000000000000ef
0x000000000000000000000000000000000000000000000000000000000000beef
0x0000000000000000000000000000000000000000000000000000000000adbeef
0x00000000000000000000000000000000000000000000000000000000deadbeef

                                    ...

0x0000beefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadbefe
0x0000beefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadcafe
```

Variable payload sizes to accommodate different situations

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x00000000000000000000000000000000000000000000000000000000000000ef
```

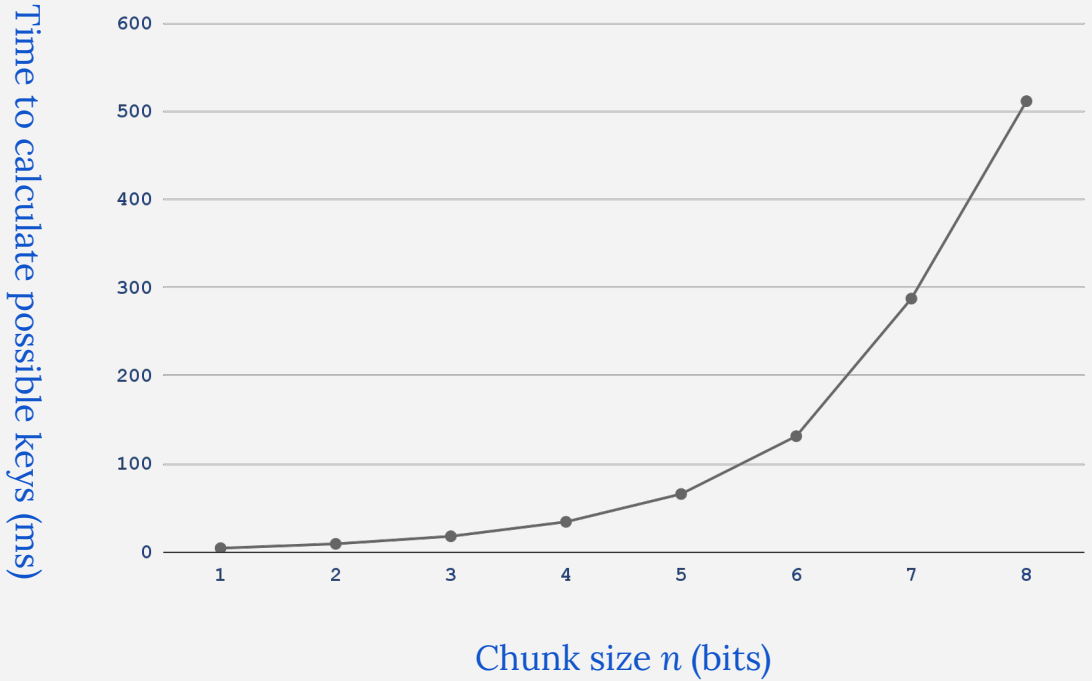
- Here, the chunk size n = 8 bits
- Need to broadcast out more packets for the same length payload
- Less computation on desktop side to decode
- Useful for areas with high traffic

Variable payload sizes to accommodate different situations

```
0x0000000000000000000000000000000000000000000000000000000000000000
0x0000beefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeefdeadbeef
```

- Here, the chunk size n = 208 bits
- Need to broadcast out fewer packets for the same length payload
- More computation on desktop side to decode
- Useful for areas with low traffic, only one chance at connection

Variable payload sizes to accommodate different situations



More efficient usage of BLE advertisement packet

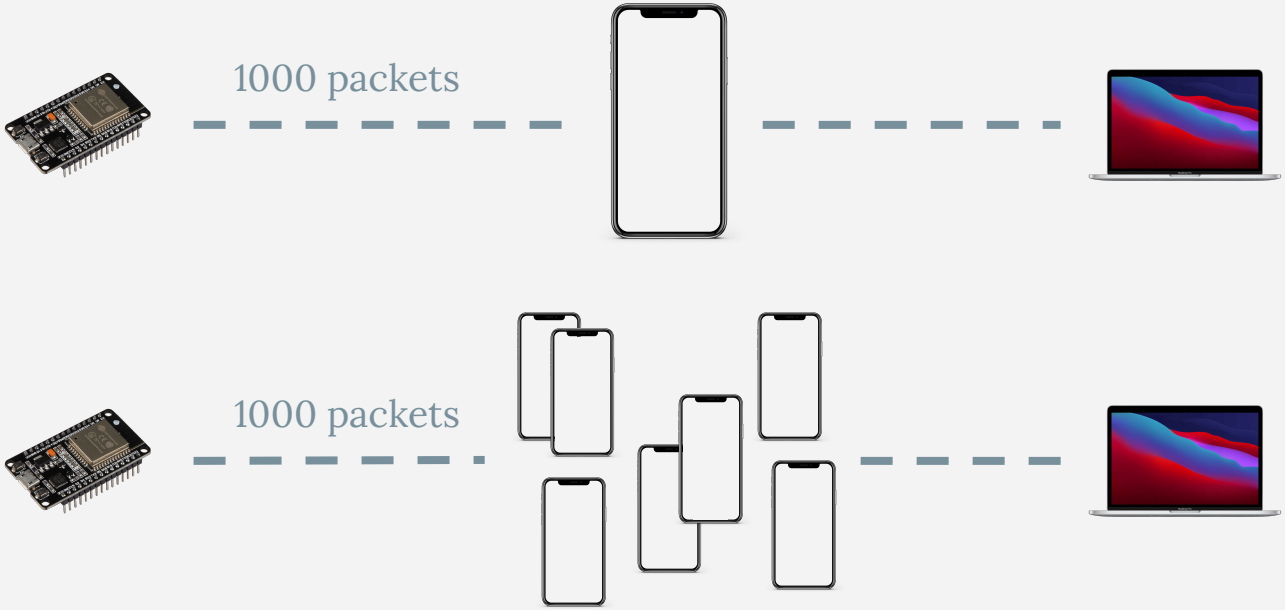
```
00000008 00000002 deadbeef 00000002 00000000000000000000 01
bit index     message ID      modem ID      counter               padding 0s            bit

0002 00000000000000000000000000000000000000000000000000000001
counter                                payload
```

- Original broadcasting speed of 1 bit per packet: **3 bytes/s**
- TagAlong, with chunk size n = 8 bits: **12.5 bytes/s**
  - For the maximum 208 bit chunk size: **325 bytes/s**

Throughput is especially important for TagAlong, no ACKs

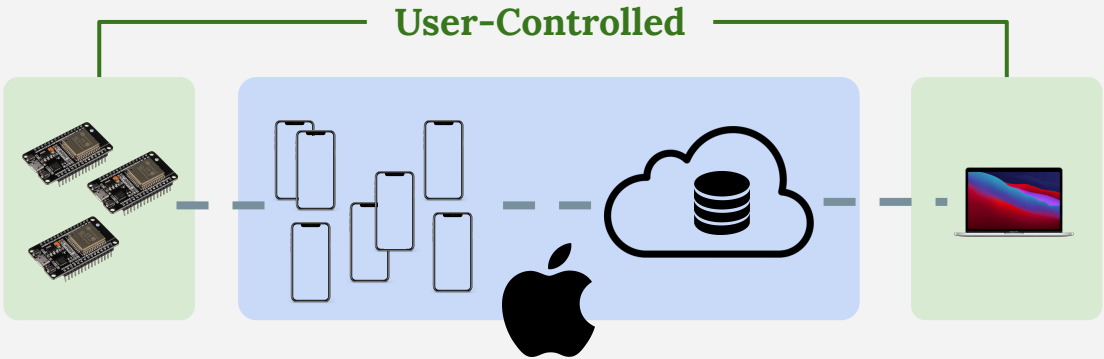
Throughput is especially important for TagAlong, no ACKs



	Each packet **sent 1x**	Each packet **sent 2x**	Each packet **sent 5x**
Percent received, **isolated location**	5.7%	51.2%	42.2%
Percent received, **busy location**	65.7%	73.2%	94.7%

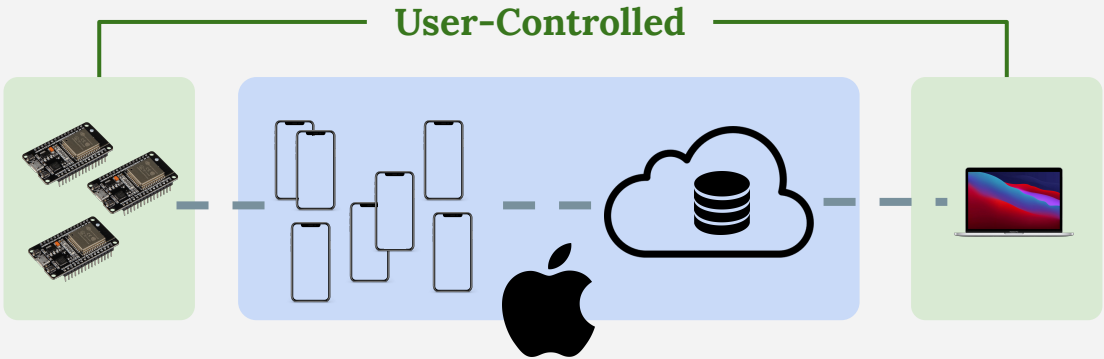
TagAlong

- Arbitrary data transmission over Apple's Find My protocol
- No infrastructure to deploy, uses nearby Apple devices to carry data
- 12 bytes per second output data rate, with up to 97% data reception



TagAlong

- Arbitrary data transmission over Apple's Find My protocol
- No infrastructure to deploy, uses nearby Apple devices to carry data
- 12 bytes per second output data rate, with up to 97% data reception



Questions?