USENIX Security

August 14th, 2019

A Study of the Feasibility of Co-located App Attacks against BLE

and a

Large-Scale Analysis of the Current Application-Layer Security Landscape

Pallavi Sivakumaran, Jorge Blasco

EPSRC
Engineering and Physical Sciences Research Council

Centre for Doctoral Training in Cyber Security
Information Security Group
Royal Holloway University of London

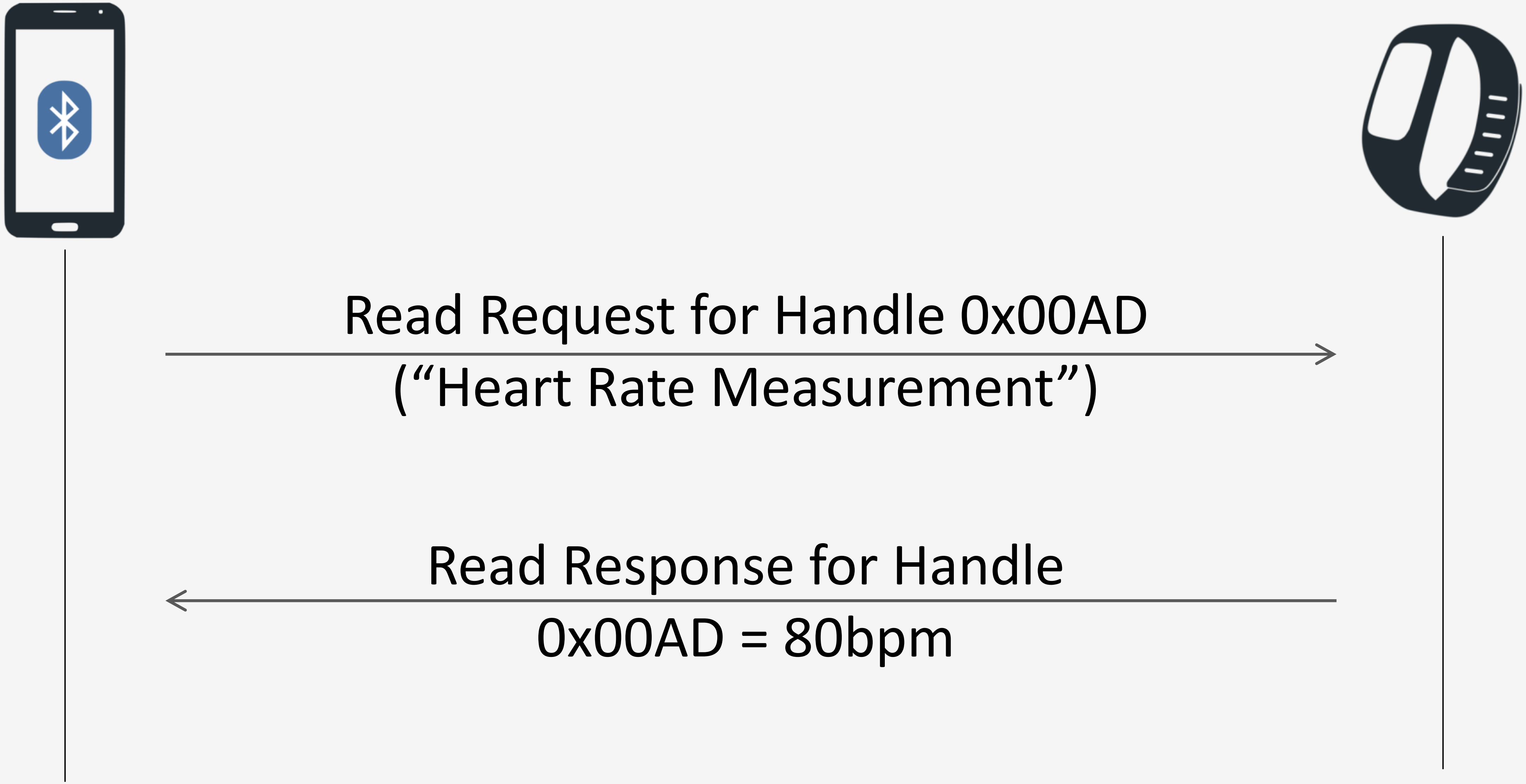
Background: Bluetooth Low Energy Data Access and Pairing

Attributes

Handle	Attribute Type	Attribute Value
…	…	…
0x00AB	Primary Service	Heart Rate Service
0x00AC	Characteristic	Heart Rate Measurement
0x00AD	Heart Rate Measurement	80bpm
0x00AE	Characteristic	Heart Rate Control Point
0x00AF	Heart Rate Control Point	
…	…	…

Attributes

Handle	Attribute Type	Attribute Value
…	…	…
0x00AB	Primary Service	Heart Rate Service
0x00AC	Characteristic	Heart Rate Measurement
0x00AD	Heart Rate Measurement	80bpm
0x00AE	Characteristic	Heart Rate Control Point
0x00AF	Heart Rate Control Point	
…	…	…


Read Request for Handle 0x00AD
("Heart Rate Measurement")
Read Response for Handle
0x00AD = 80bpm

0x0005	Lock status	0x01

0x00AD	Heart Rate Measurement	80bpm

0x001D	Glucose Measurement	135mg/dL

0x002E	Travel speed	70kmph

0x0011	Gas valve	0x01

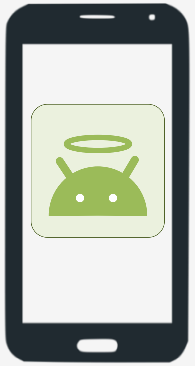
- Permissions
  - Access
  - Authentication (pairing)
  - Authorization

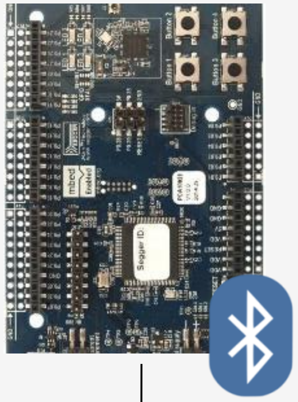

135

Q1: Can an Unauthorised App Access Protected Data?


mbed Enabled
Segger ID:

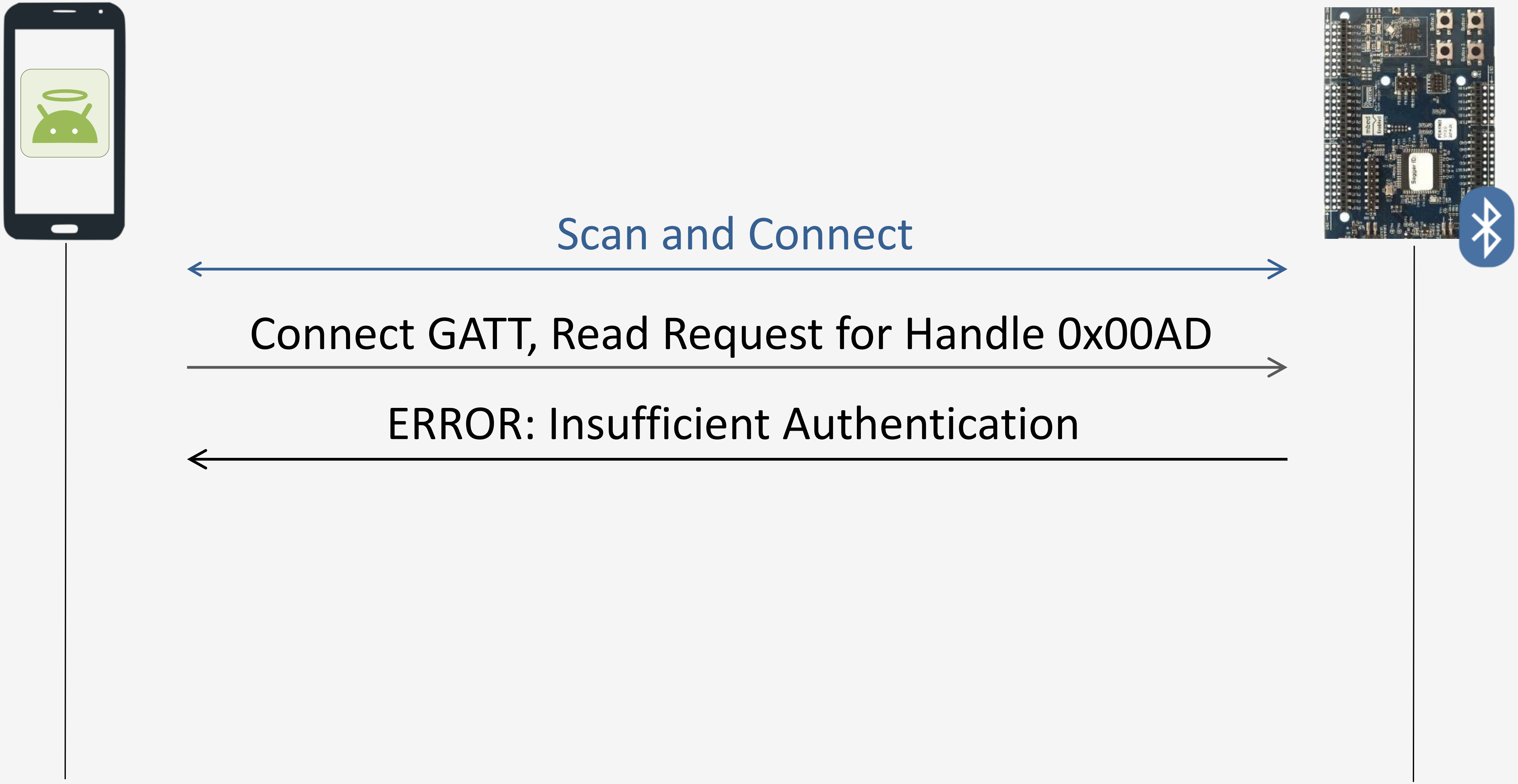
Co-located App
Data Access Scenario #1

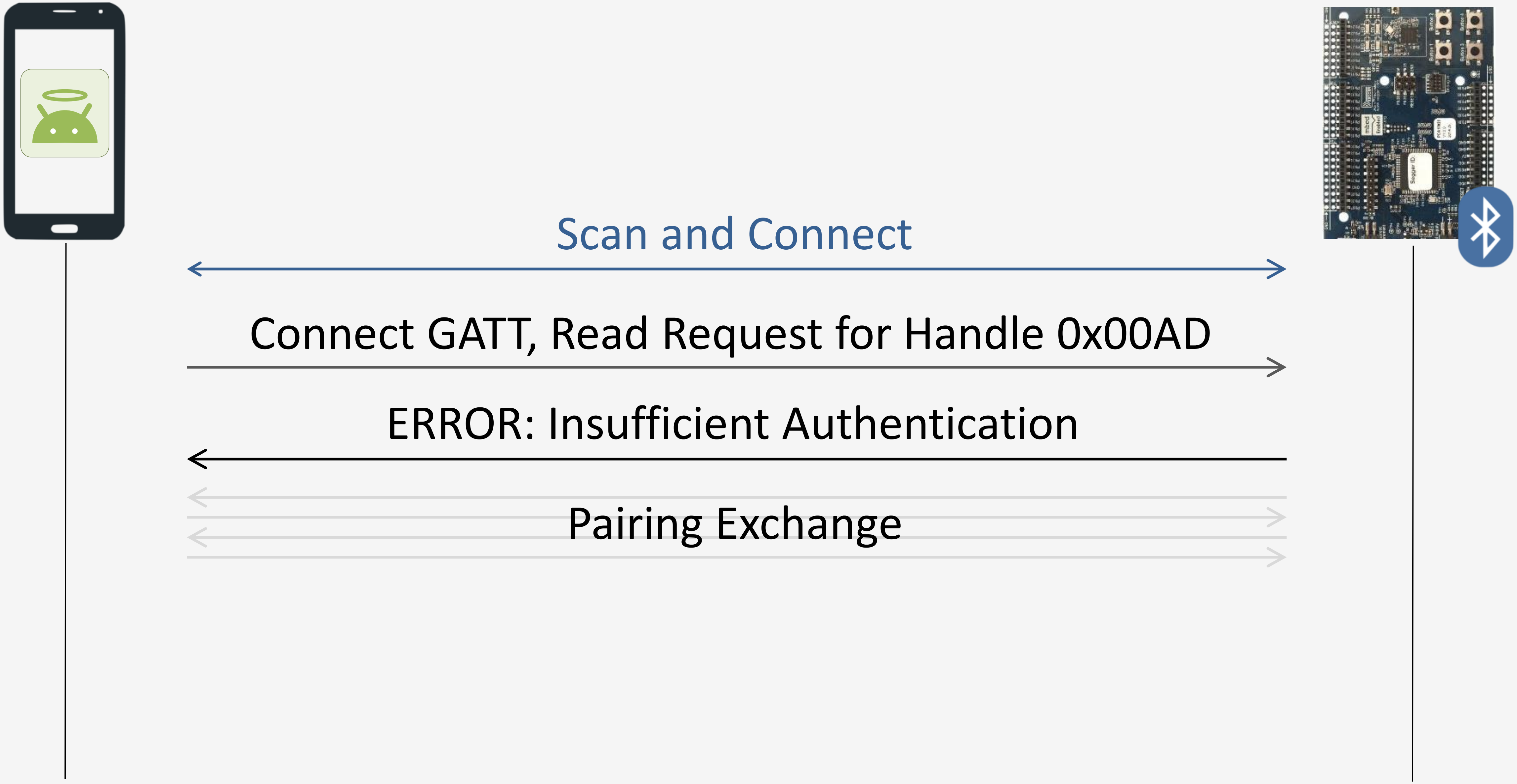


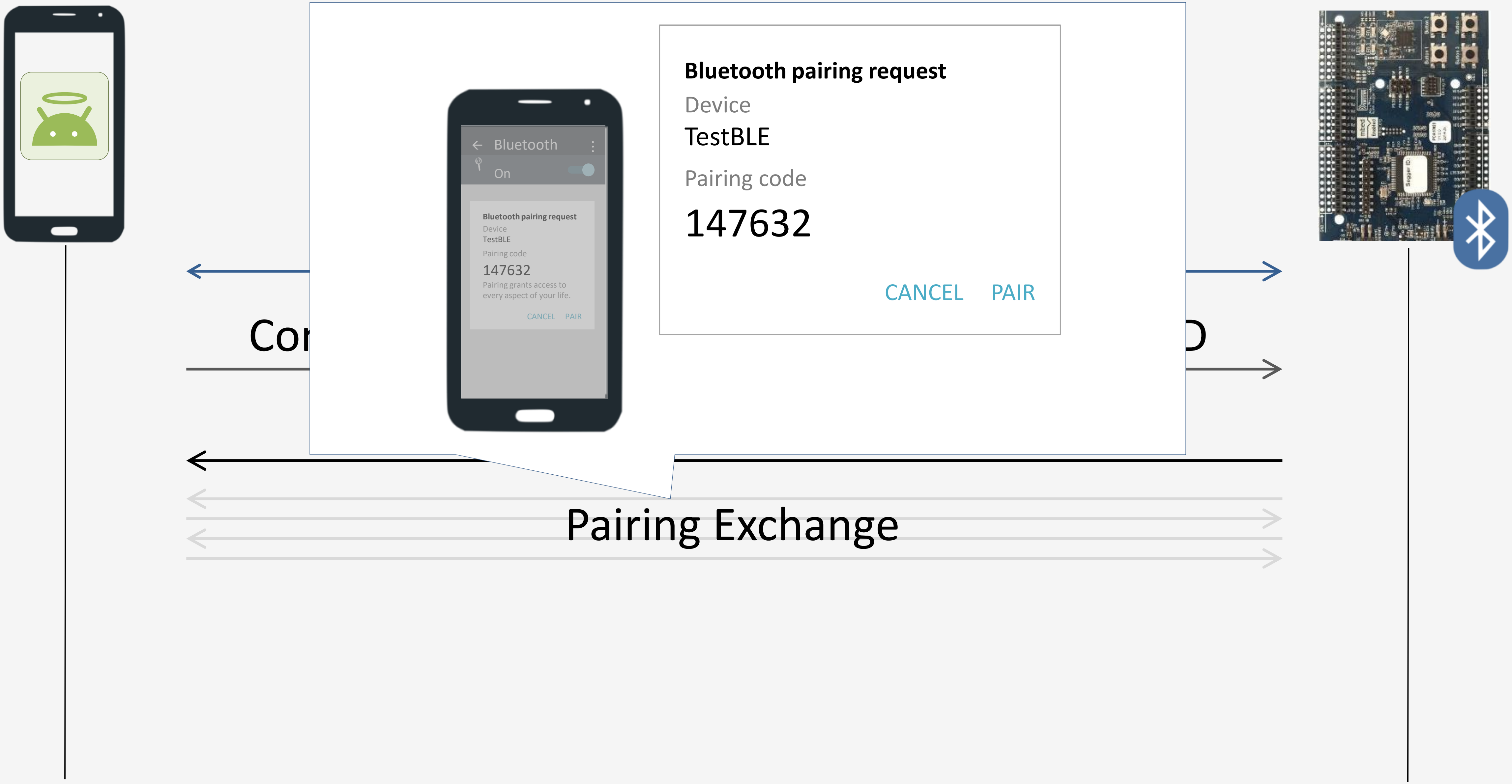


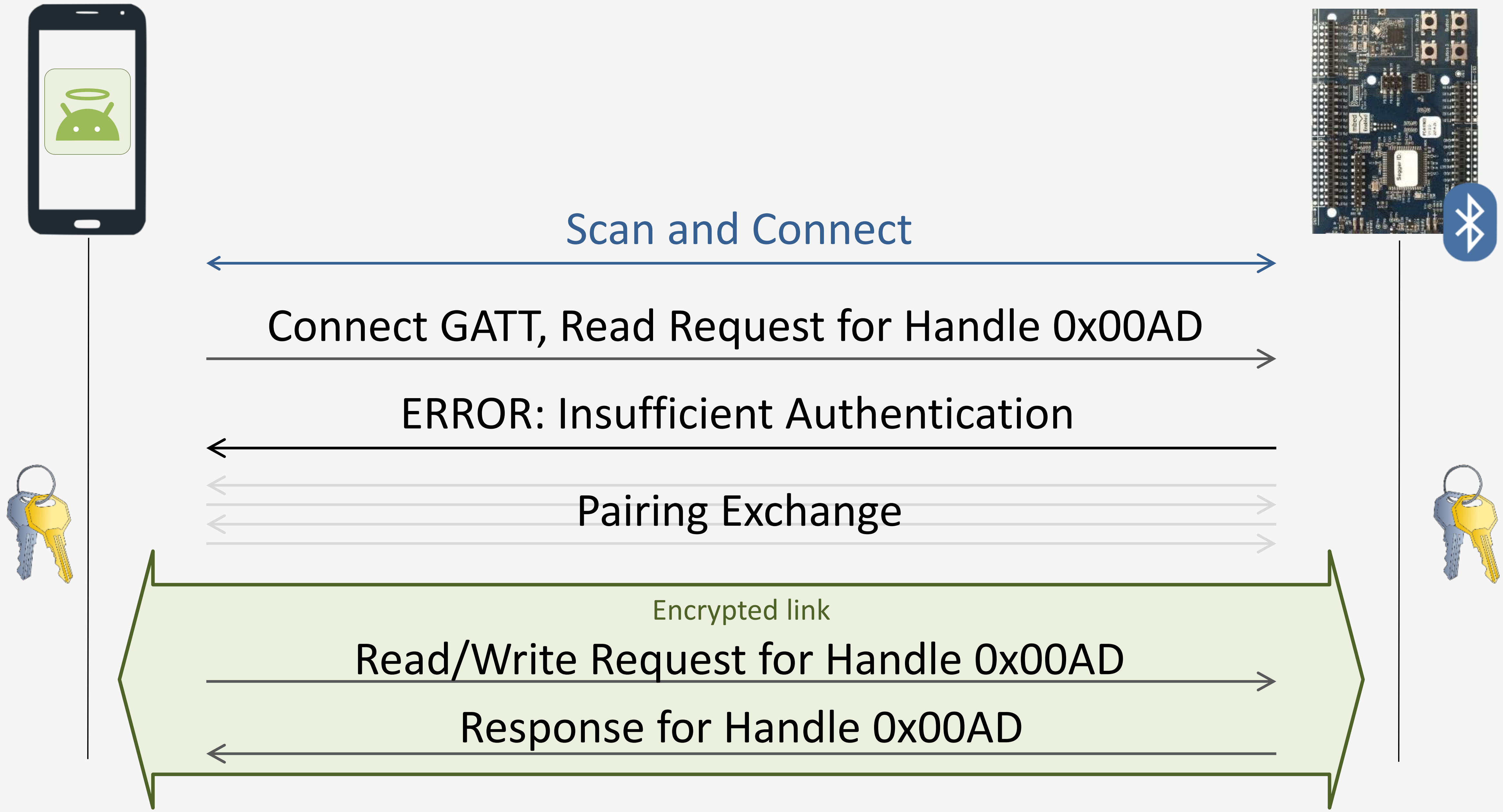
Scan and Connect

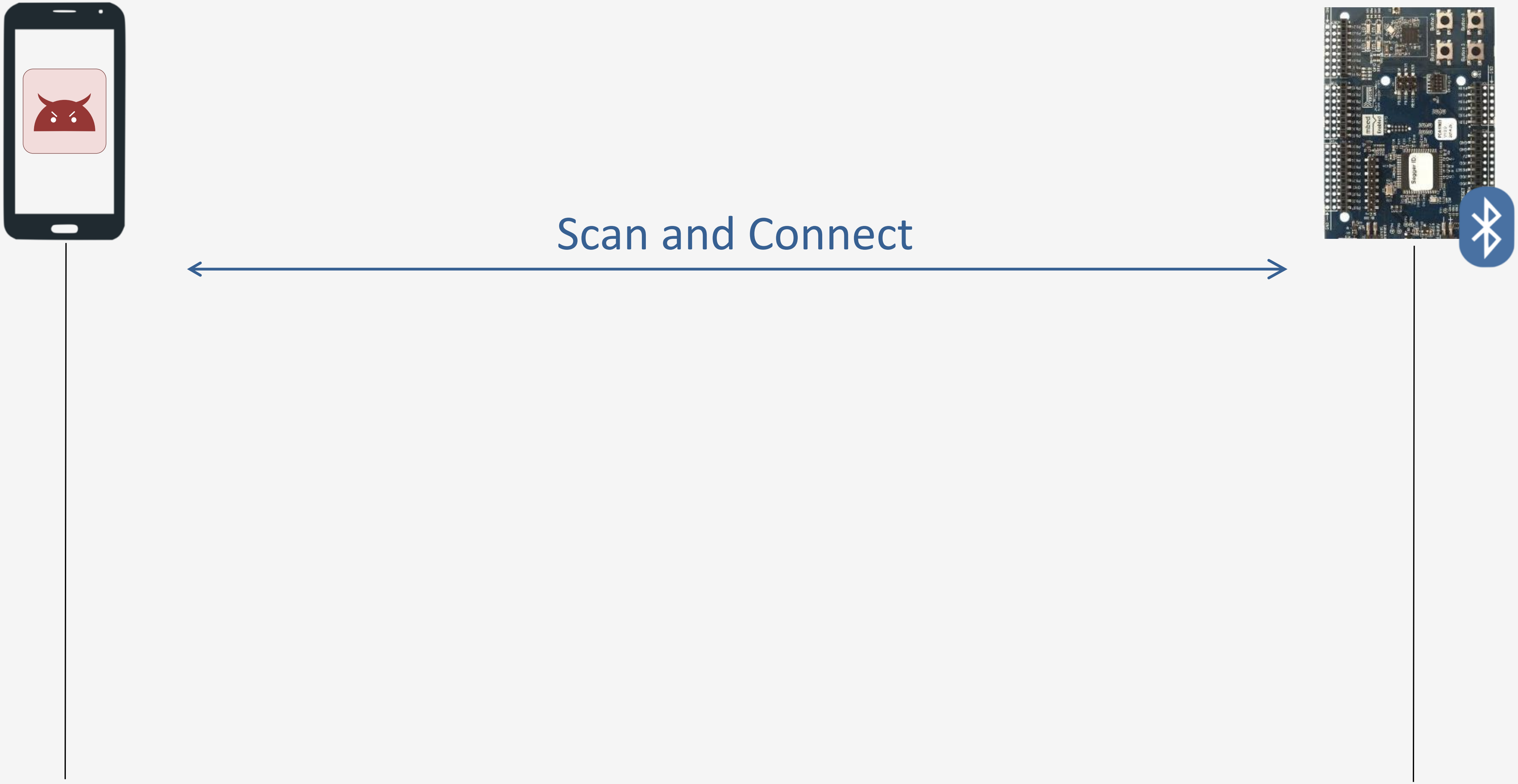
Connect GATT, Read Request for Handle 0x00AD

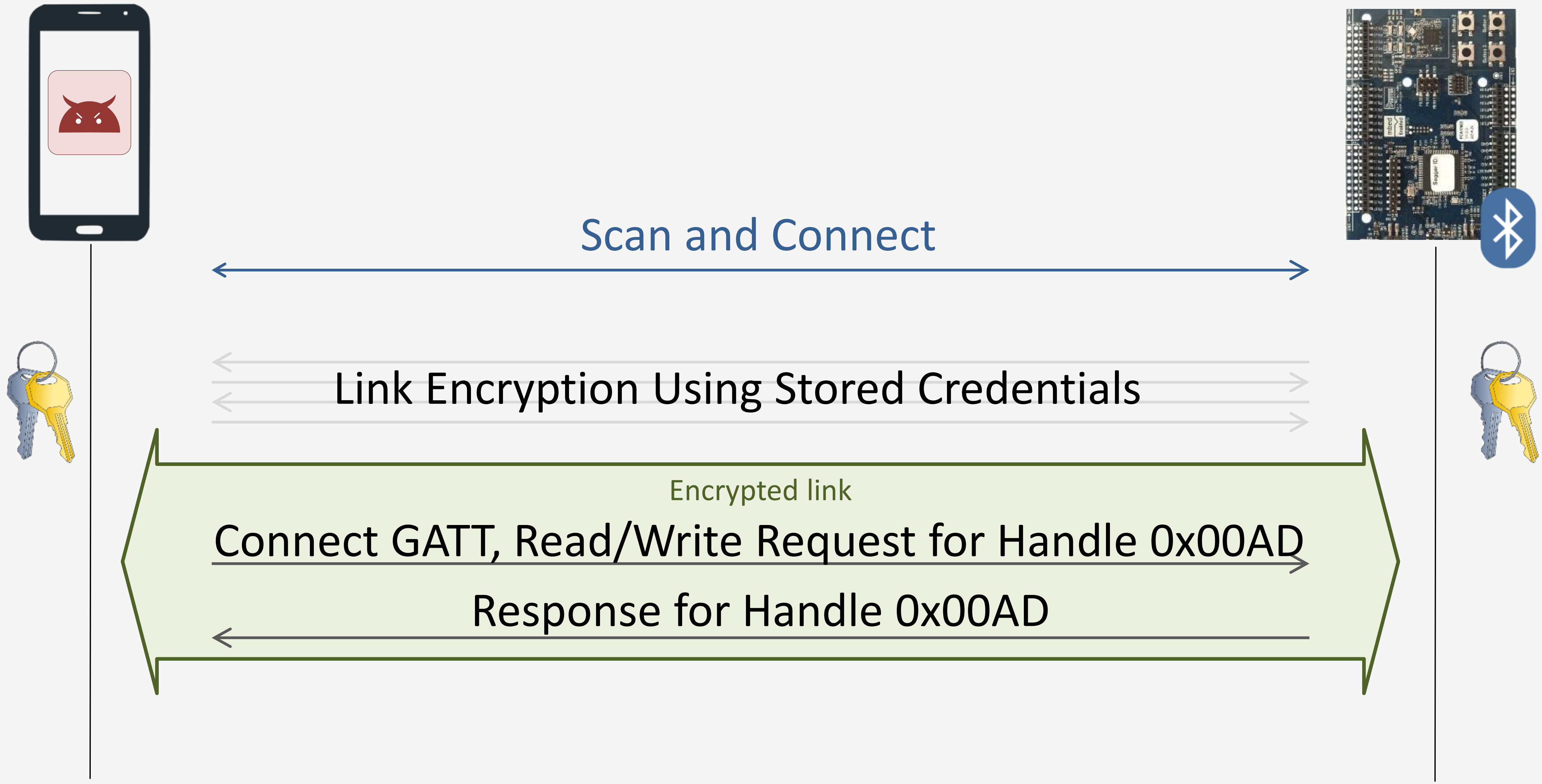

Scan and Connect
Connect GATT, Read Request for Handle 0x00AD
ERROR: Insufficient Authentication

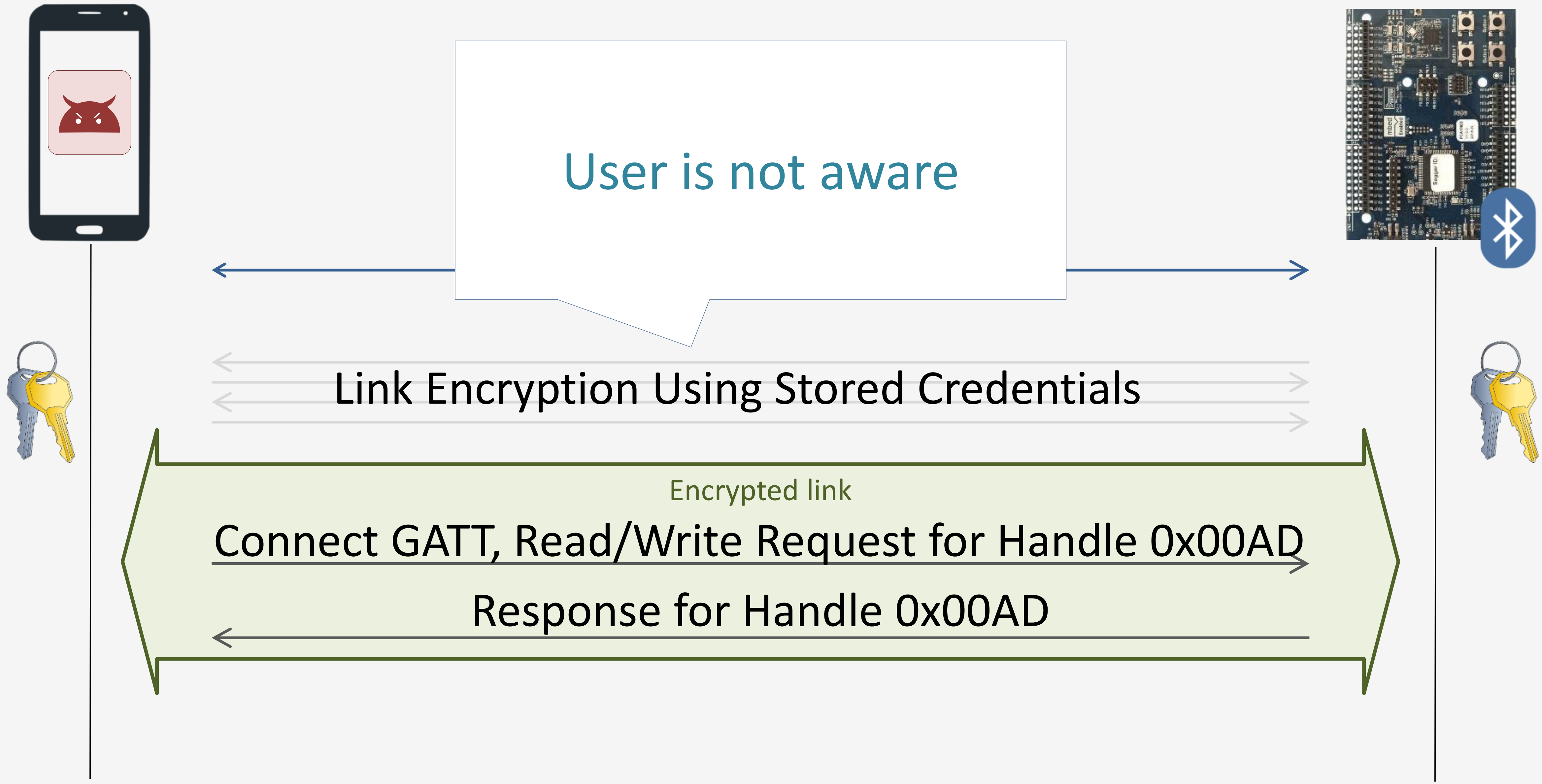

Scan and Connect
Connect GATT, Read Request for Handle 0x00AD
ERROR: Insufficient Authentication
Pairing Exchange

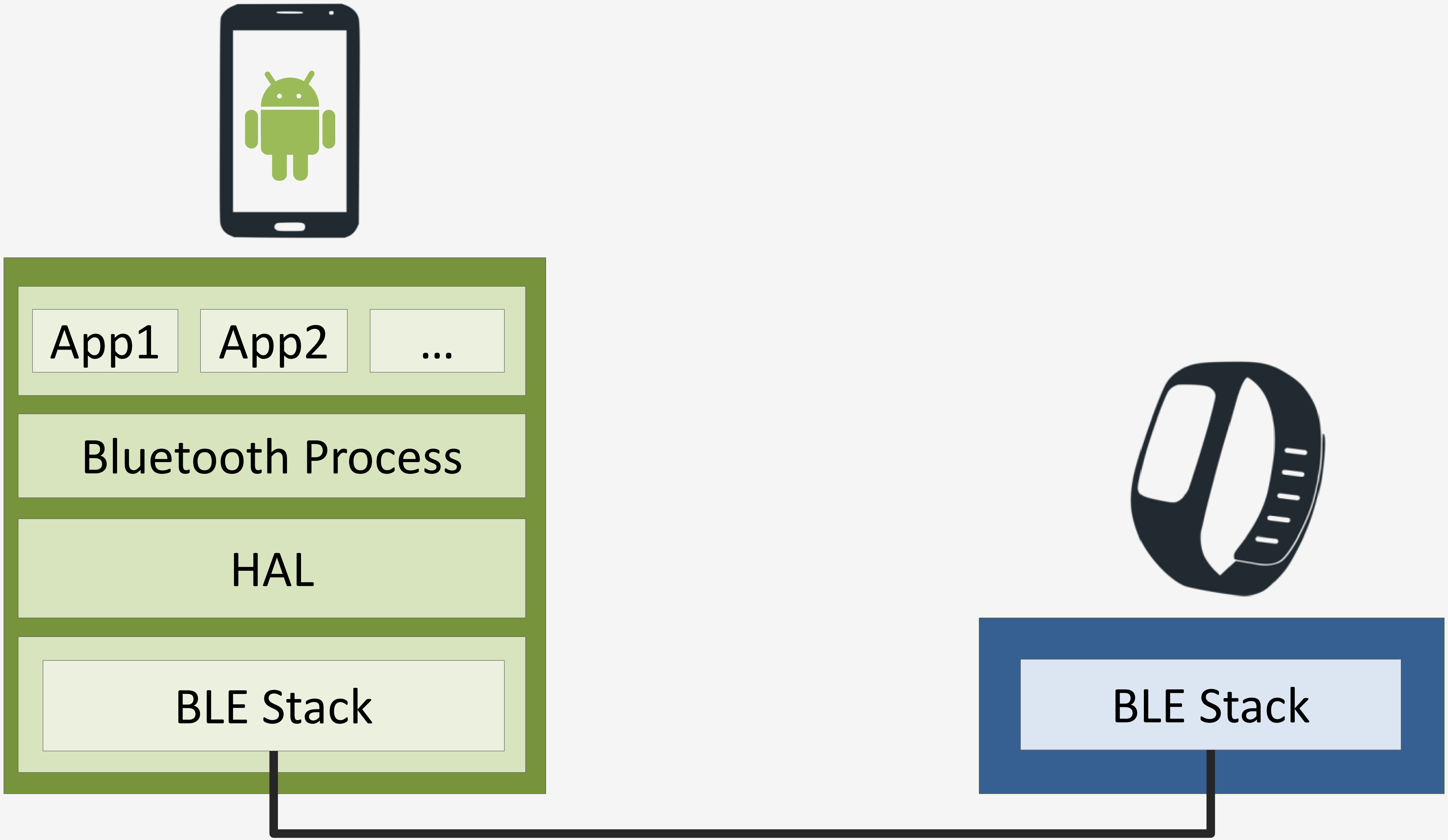

Bluetooth
On
Bluetooth pairing request
Device
TestBLE
Pairing code
147632
Pairing grants access to every aspect of your life.
CANCEL
PAIR
Bluetooth pairing request
Device
TestBLE
Pairing code
147632
CANCEL
PAIR
Pairing Exchange

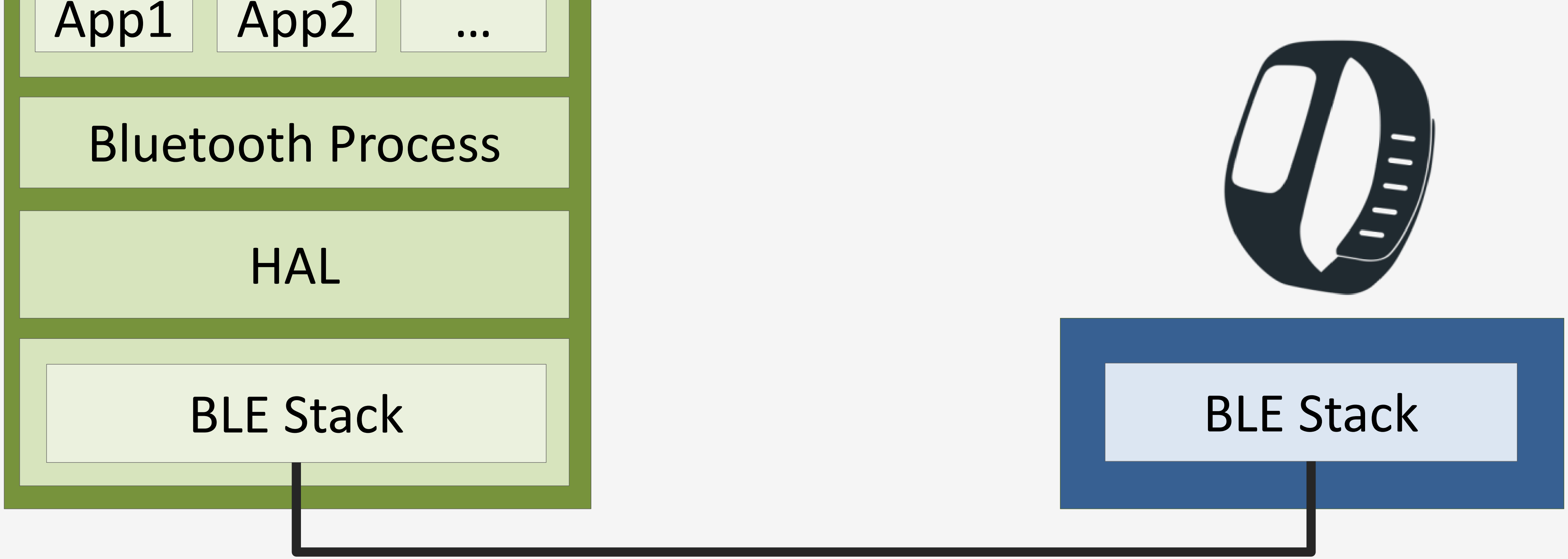

Scan and Connect
Connect GATT, Read Request for Handle 0x00AD
ERROR: Insufficient Authentication
Pairing Exchange
Encrypted link
Read/Write Request for Handle 0x00AD
Response for Handle 0x00AD

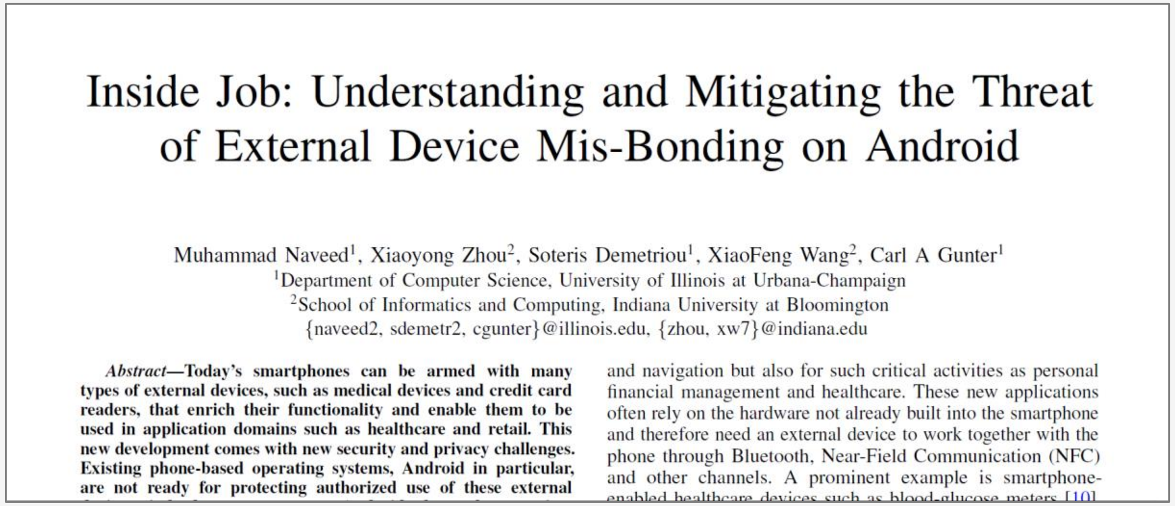

Scan and Connect


Scan and Connect
Link Encryption Using Stored Credentials
Encrypted link
Connect GATT, Read/Write Request for Handle 0x00AD
Response for Handle 0x00AD


User is not aware
Link Encryption Using Stored Credentials
Encrypted link
Connect GATT, Read/Write Request for Handle 0x00AD
Response for Handle 0x00AD


App1
App2
...
Bluetooth Process
HAL
BLE Stack
BLE Stack





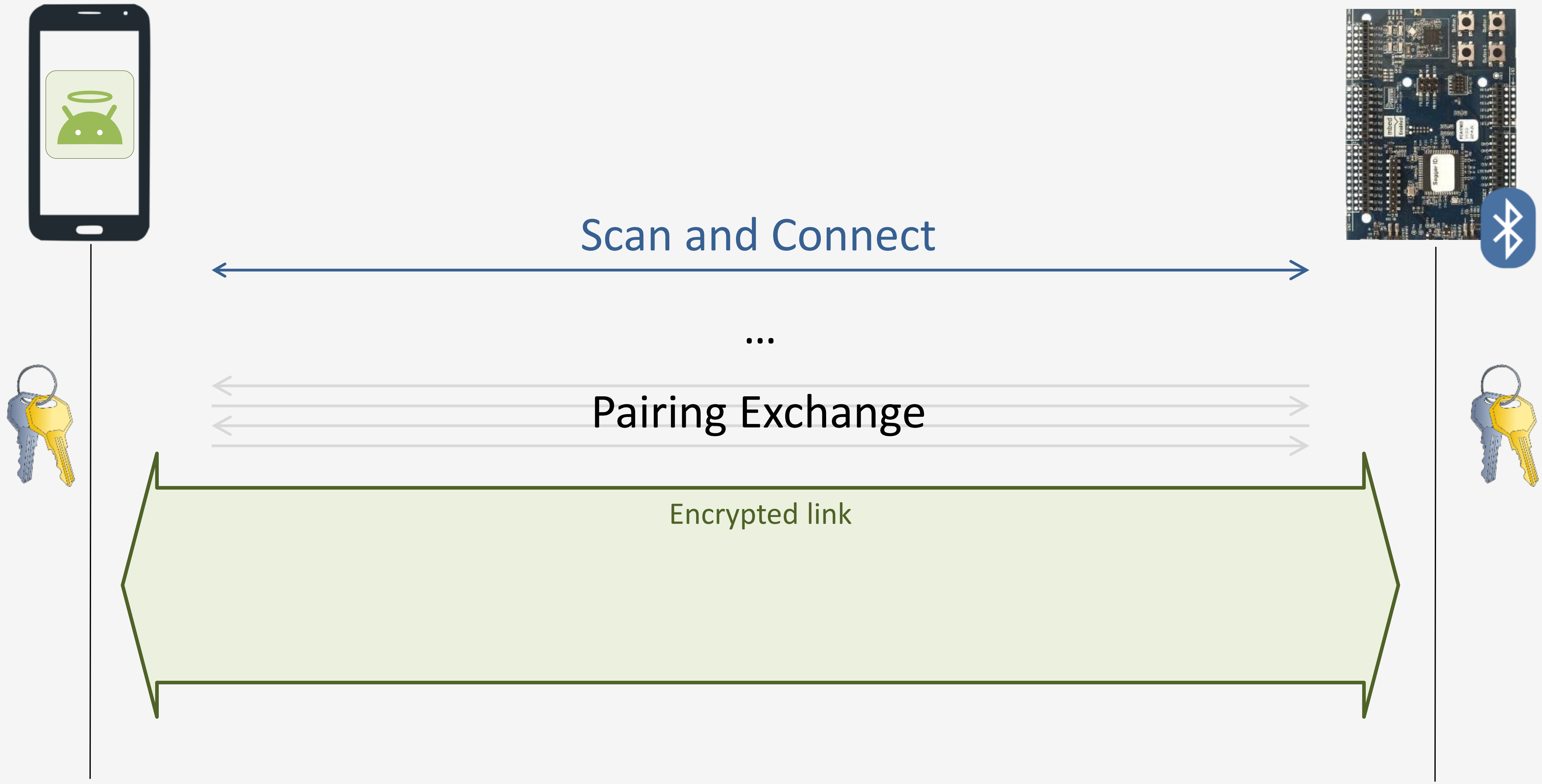
Inside Job: Understanding and Mitigating the Threat of External Device Mis-Bonding on Android

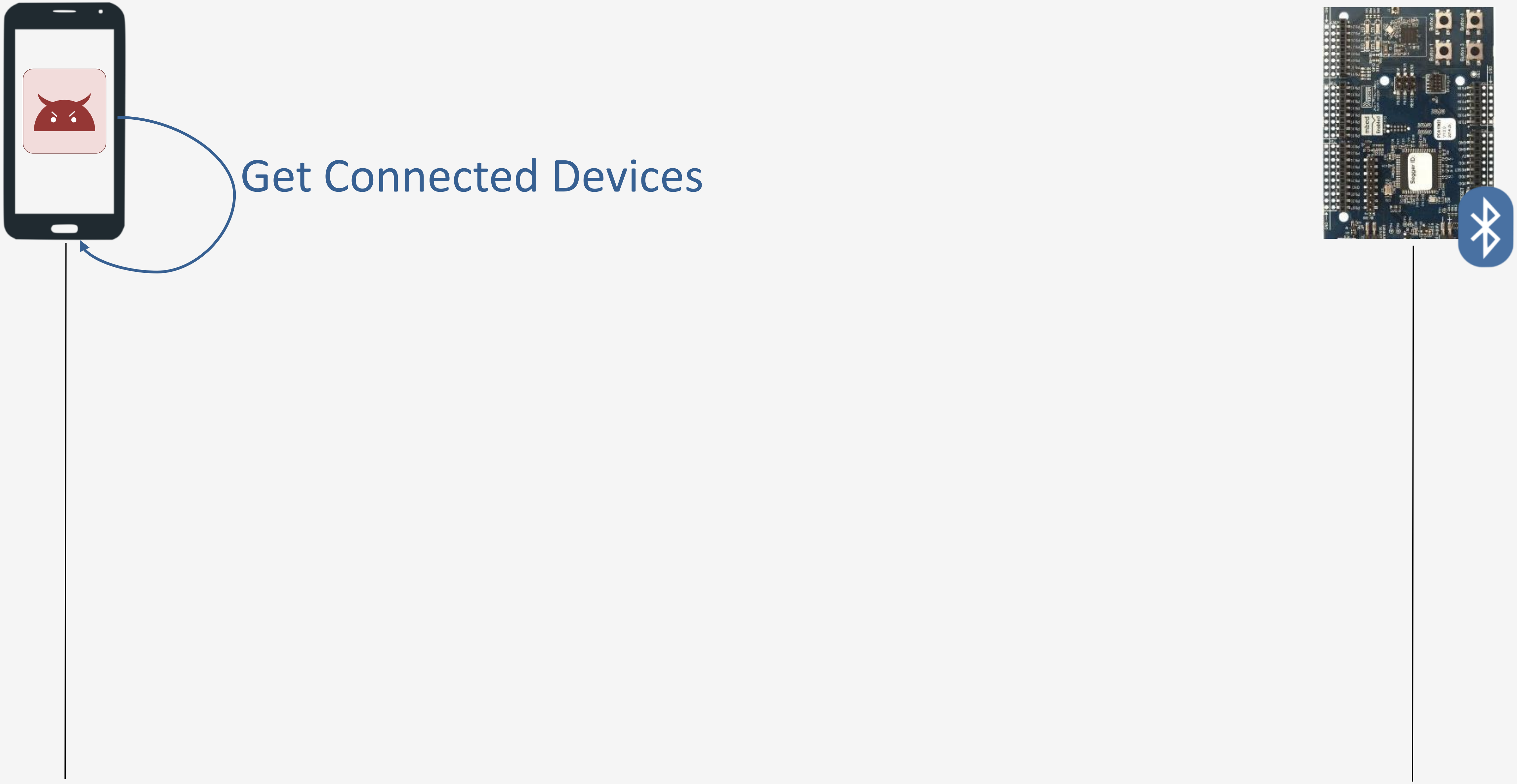
Muhammad Naveed[1], Xiaoyong Zhou[2], Soteris Demetriou[1], XiaoFeng Wang[2], Carl A Gunter[1]
[1]Department of Computer Science, University of Illinois at Urbana-Champaign
[2]School of Informatics and Computing, Indiana University at Bloomington
{naveed2, sdemetr2, cgunter}@illinois.edu, {zhou, xw7}@indiana.edu

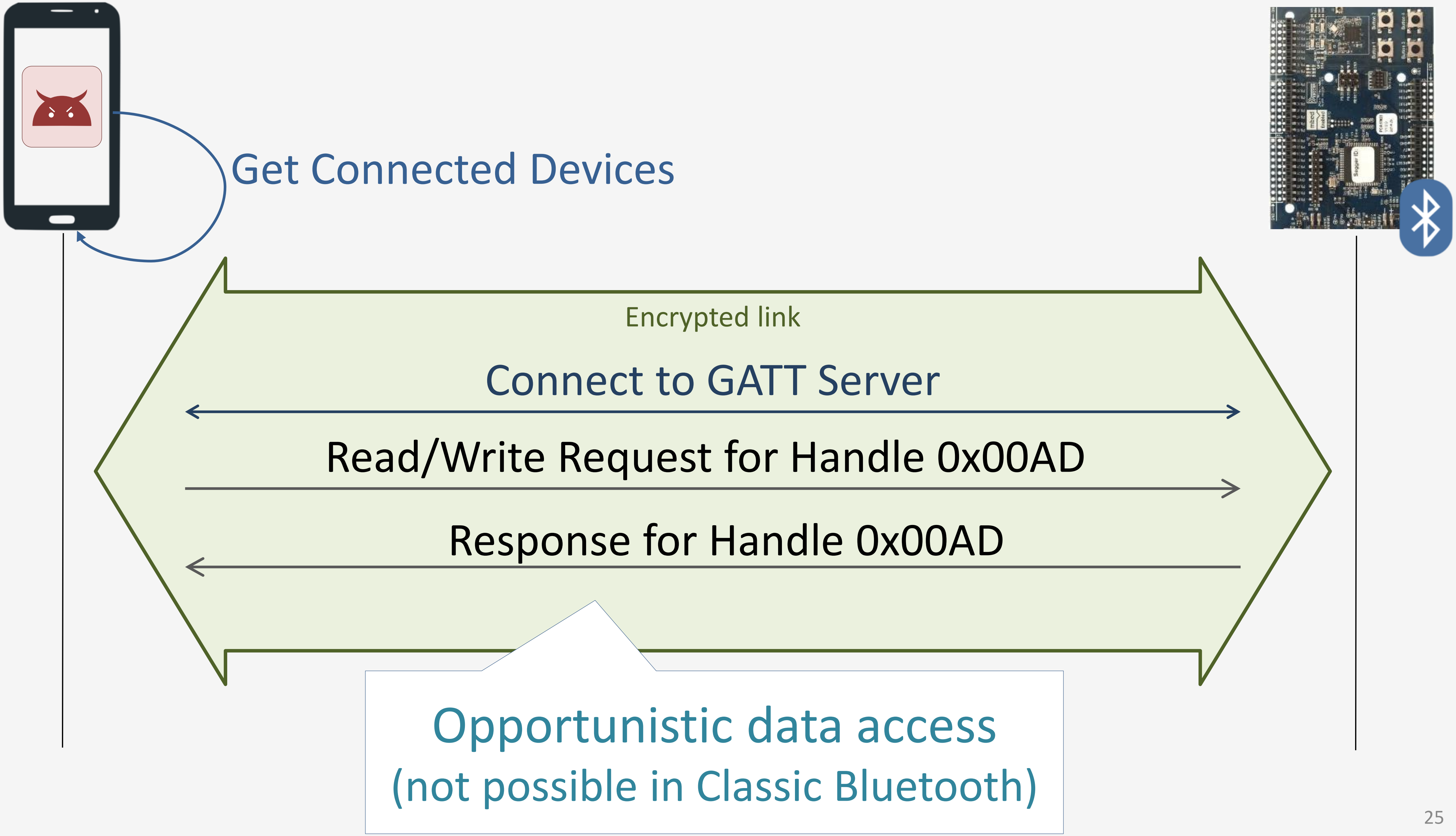
***Abstract*—Today's smartphones can be armed with many types of external devices, such as medical devices and credit card readers, that enrich their functionality and enable them to be used in application domains such as healthcare and retail. This new development comes with new security and privacy challenges. Existing phone-based operating systems, Android in particular, are not ready for protecting authorized use of these external**

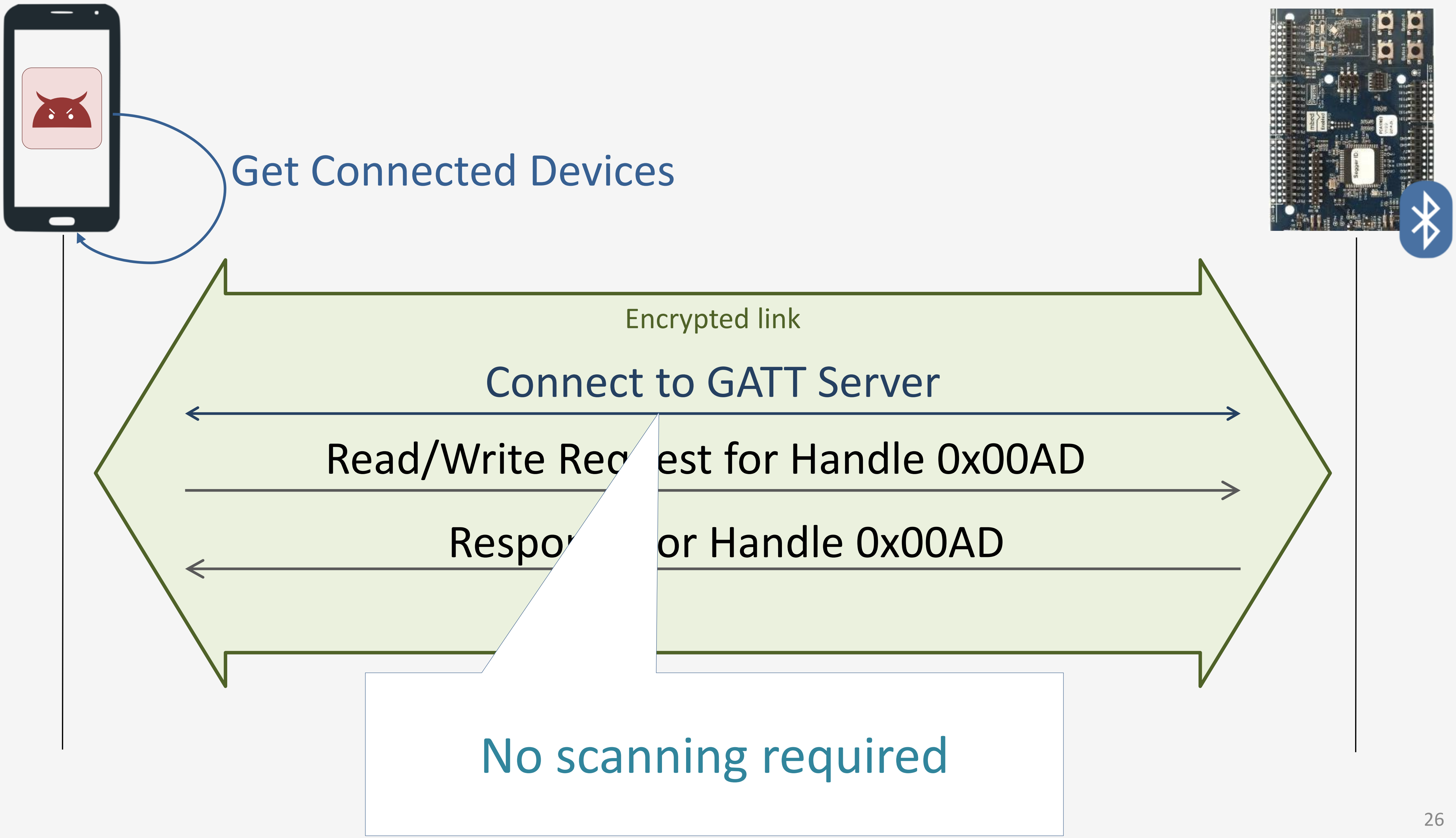
and navigation but also for such critical activities as personal financial management and healthcare. These new applications often rely on the hardware not already built into the smartphone and therefore need an external device to work together with the phone through Bluetooth, Near-Field Communication (NFC) and other channels. A prominent example is smartphone-enabled healthcare devices such as blood-glucose meters [10]

Co-located App
Data Access Scenario #2


Scan and Connect
...
Pairing Exchange
Encrypted link


Get Connected Devices


Get Connected Devices
Encrypted link
Connect to GATT Server
Read/Write Request for Handle 0x00AD
Response for Handle 0x00AD
Opportunistic data access
(not possible in Classic Bluetooth)



Get Connected Devices
Encrypted link
Connect to GATT Server
Read/Write Req est for Handle 0x00AD
Respo or Handle 0x00AD
No scanning required

GoodApp

needs access to

- Bluetooth ∨
- Bluetooth Admin ∨
- Location ∨
- Internet ∨

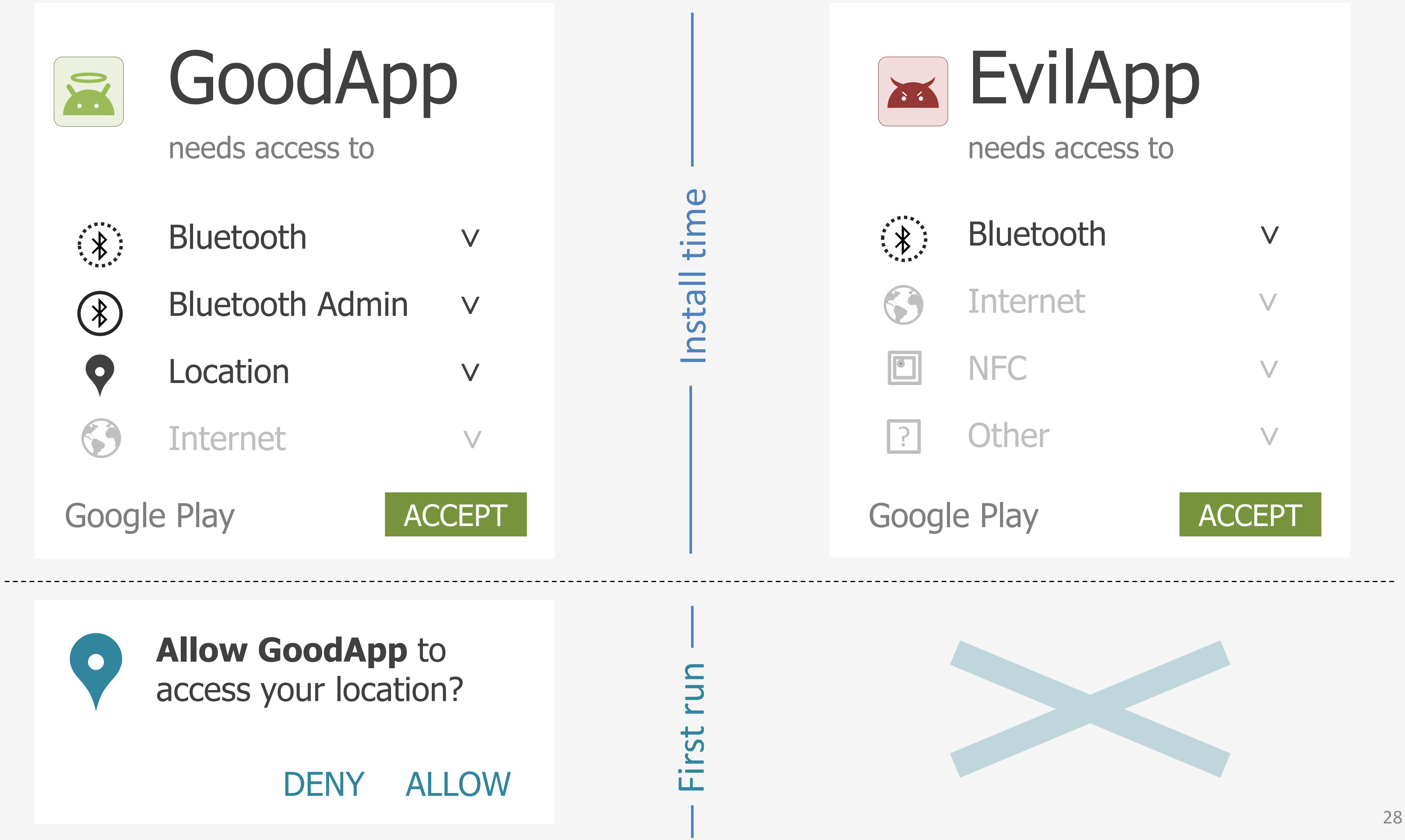
Google Play **ACCEPT**

Install time

Allow GoodApp to access your location?

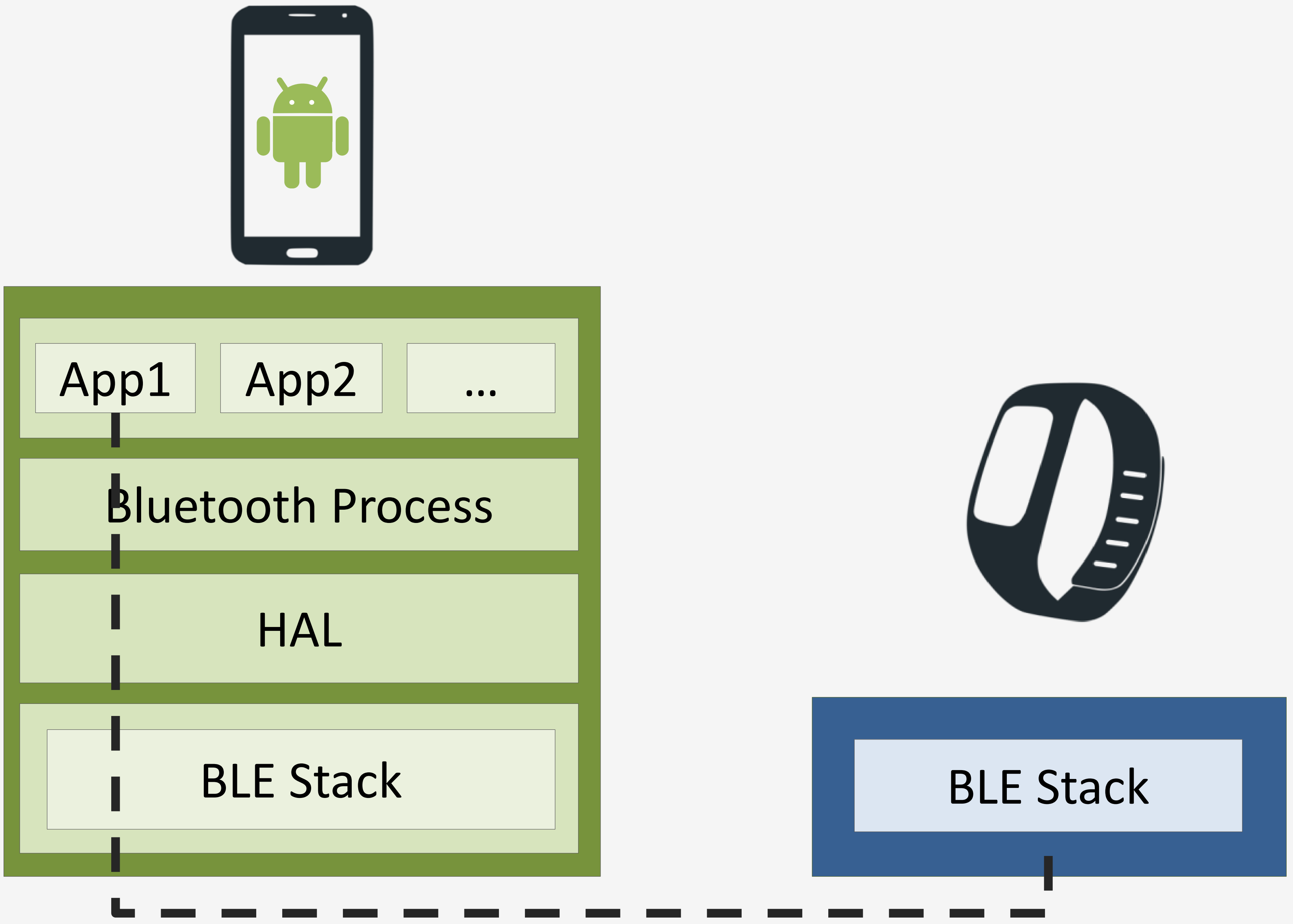
DENY ALLOW

First run


GoodApp
needs access to
Bluetooth V
Bluetooth Admin V
Location V
Internet V
Google Play
ACCEPT
Install time
EvilApp
needs access to
Bluetooth V
Internet V
NFC V
Other V
Google Play
ACCEPT
Allow GoodApp to access your location?
DENY
ALLOW
First run

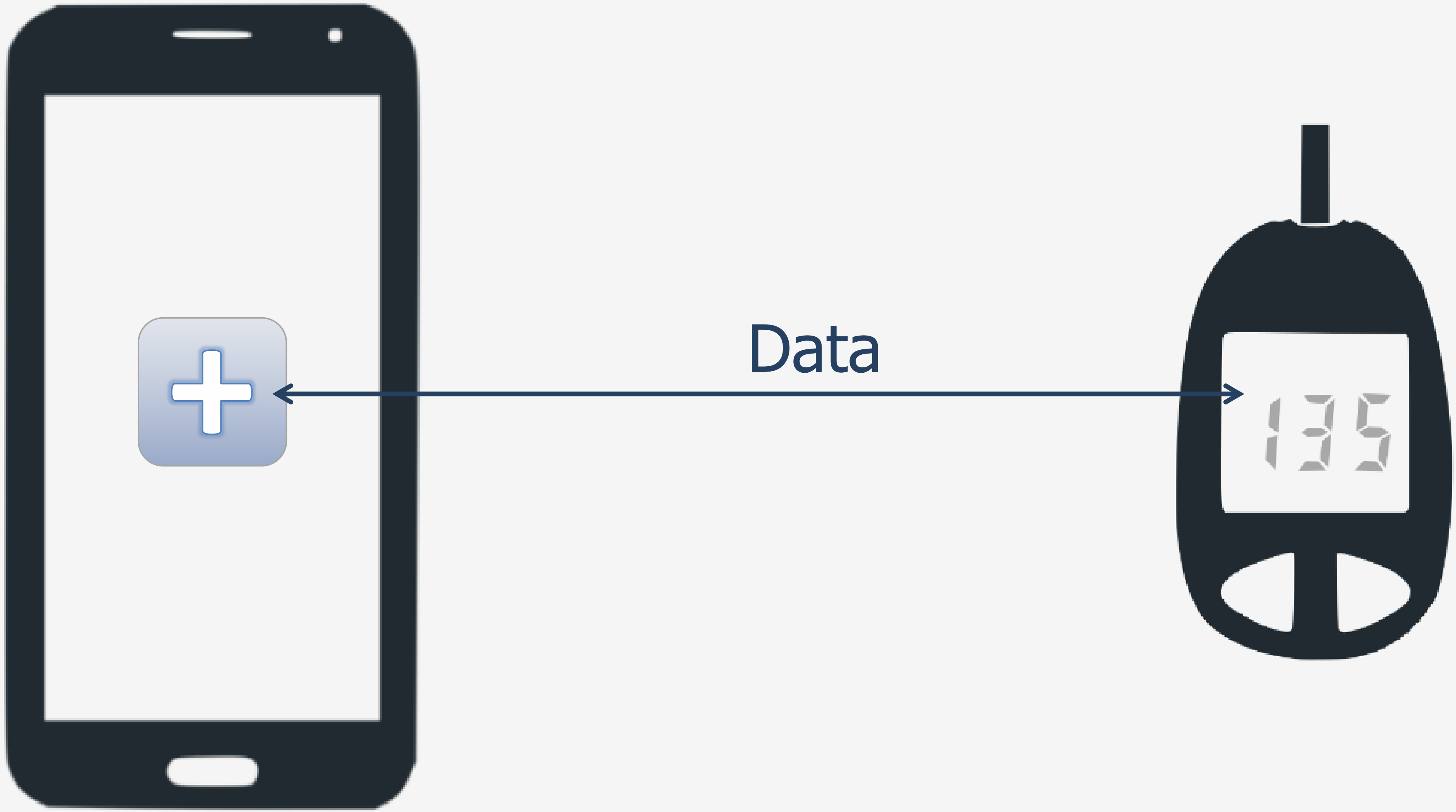
- Summary of unauthorised data access scenarios:
  - Scenario #1
    - Malicious app can access data at any time (as long as Bluetooth is on and BLE device is nearby, of course!).
    - Malicious app requires BLUETOOTH, BLUETOOTH_ADMIN, LOCATION permissions (user may view the app as being intrusive).
  - Scenario #2
    - Malicious app can only access data when good app is connected.
    - Malicious app requires only BLUETOOTH permission (activity less visible to user/app appears more benign).

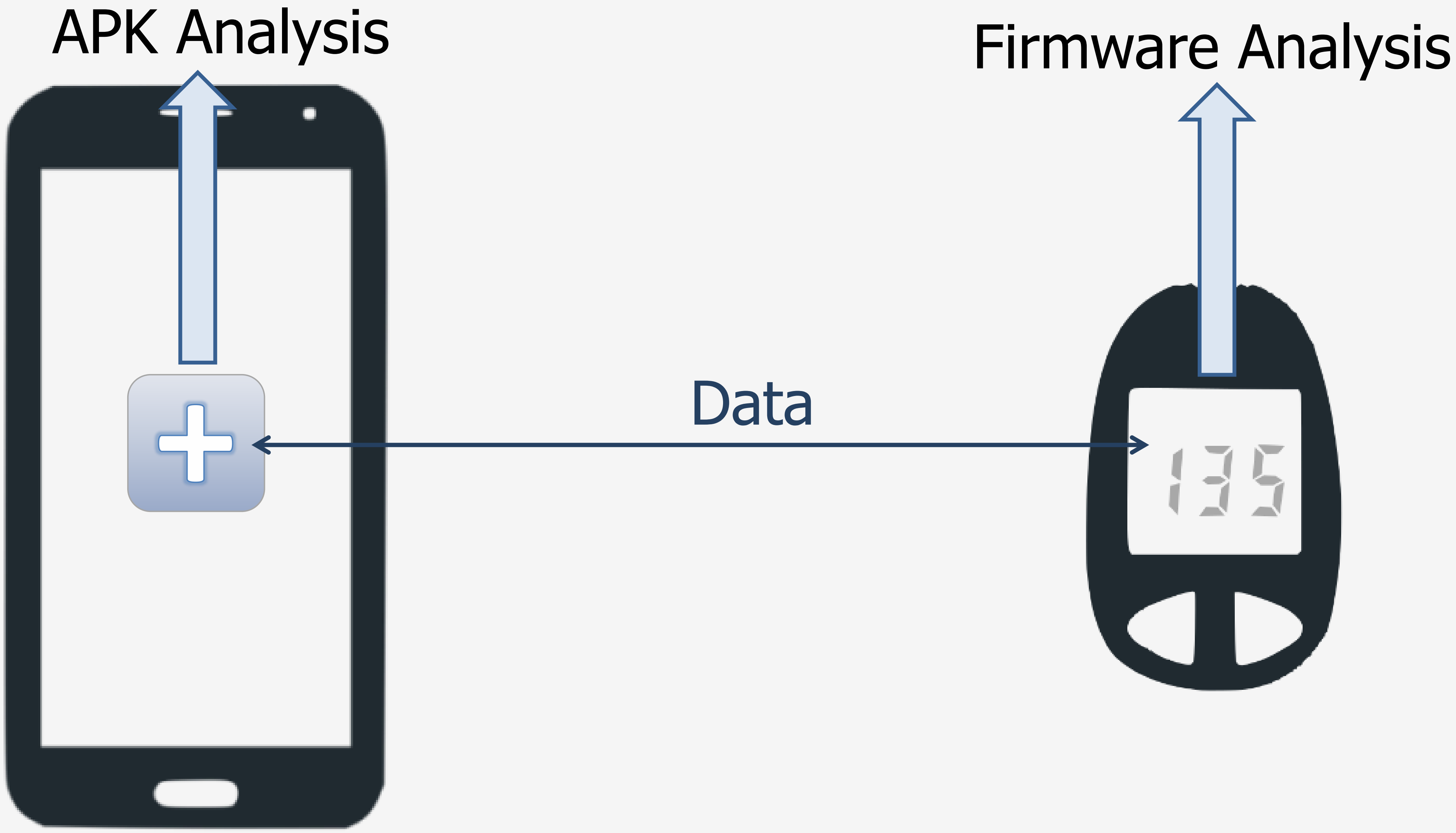
Protecting BLE Data


App1
App2
...
Bluetooth Process
HAL
BLE Stack
BLE Stack

- Several stakeholders
  - Android (and other OSs)
    - Don’t allow multiple apps to share a BLE connection.
    - Associate pairing credentials with the app that triggered pairing?
  - Bluetooth SIG
    - Add application layer protection+modify sensitive profiles. Flexibility?
  - Developers
    - Implement application-layer security ☹
    - Awareness? (We informed the Android Security Team and the Bluetooth SIG of the need for documentation regarding this issue.)

Q2: What Proportion of Devices Have End-to-End Protection for BLE Data?


Data
135


APK Analysis
Firmware Analysis
Data
135

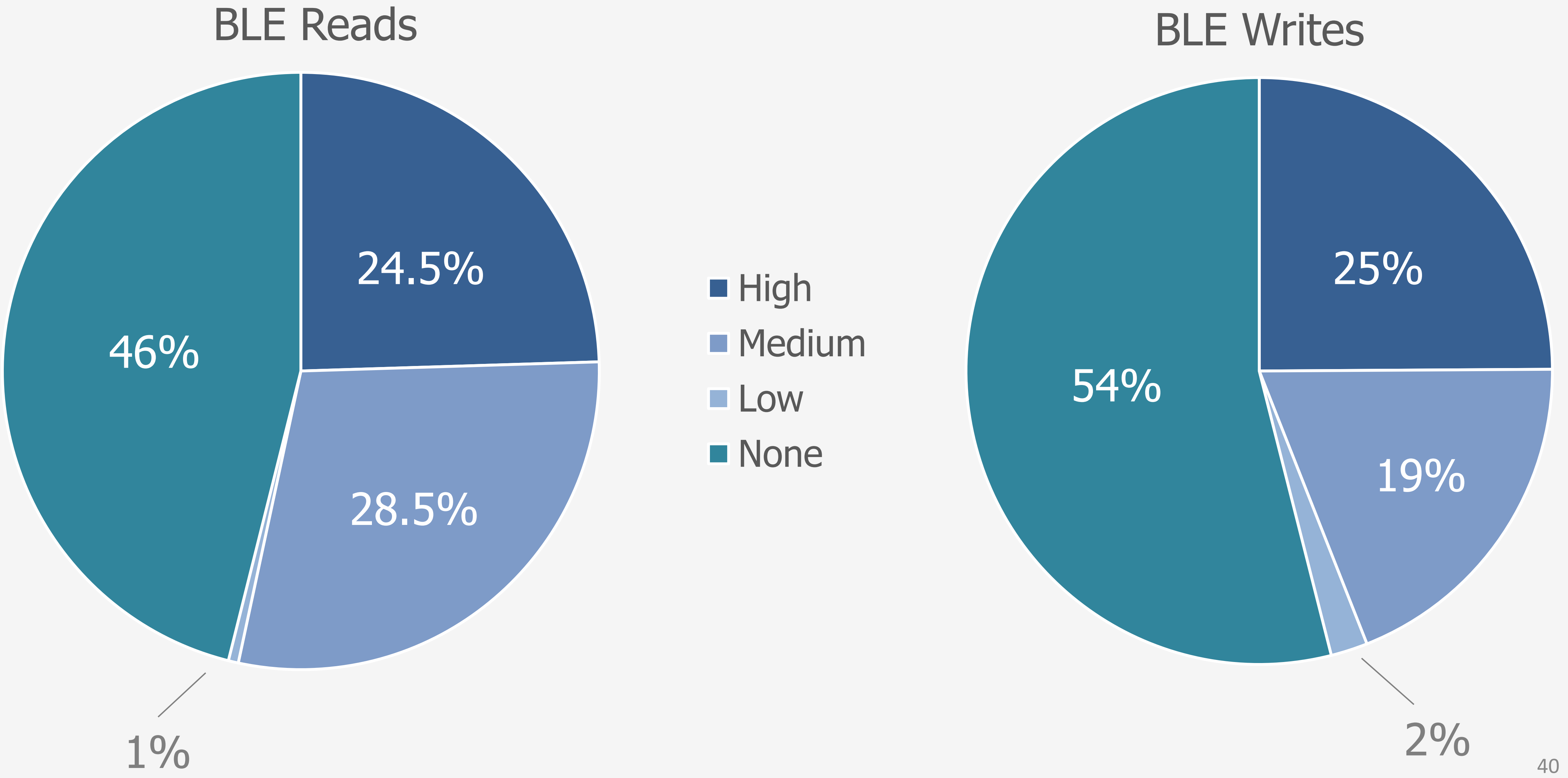
- BLECryptracer:
  - Tool to identify the presence of cryptographically-processed BLE data.
  - Analyses Android APKs:
    1. Use Androguard to obtain smali.
    2. Identify BLE data access methods.
    3. Perform “slicing” to trace through smali code, and see if we hit cryptographic libraries.

- If cryptographically-processed BLE data is identified, BLECryptracer assigns the result a "confidence level":
  - **High:** If BLE-crypto link is identified via direct register value transfers and/or immediate method invocations.
  - **Medium:** If BLE-crypto link is identified by considering abstract/interface methods and/or associated registers.
  - **Low:** If crypto is identified in any instruction within any previously encountered method (originating from BLE data access call).

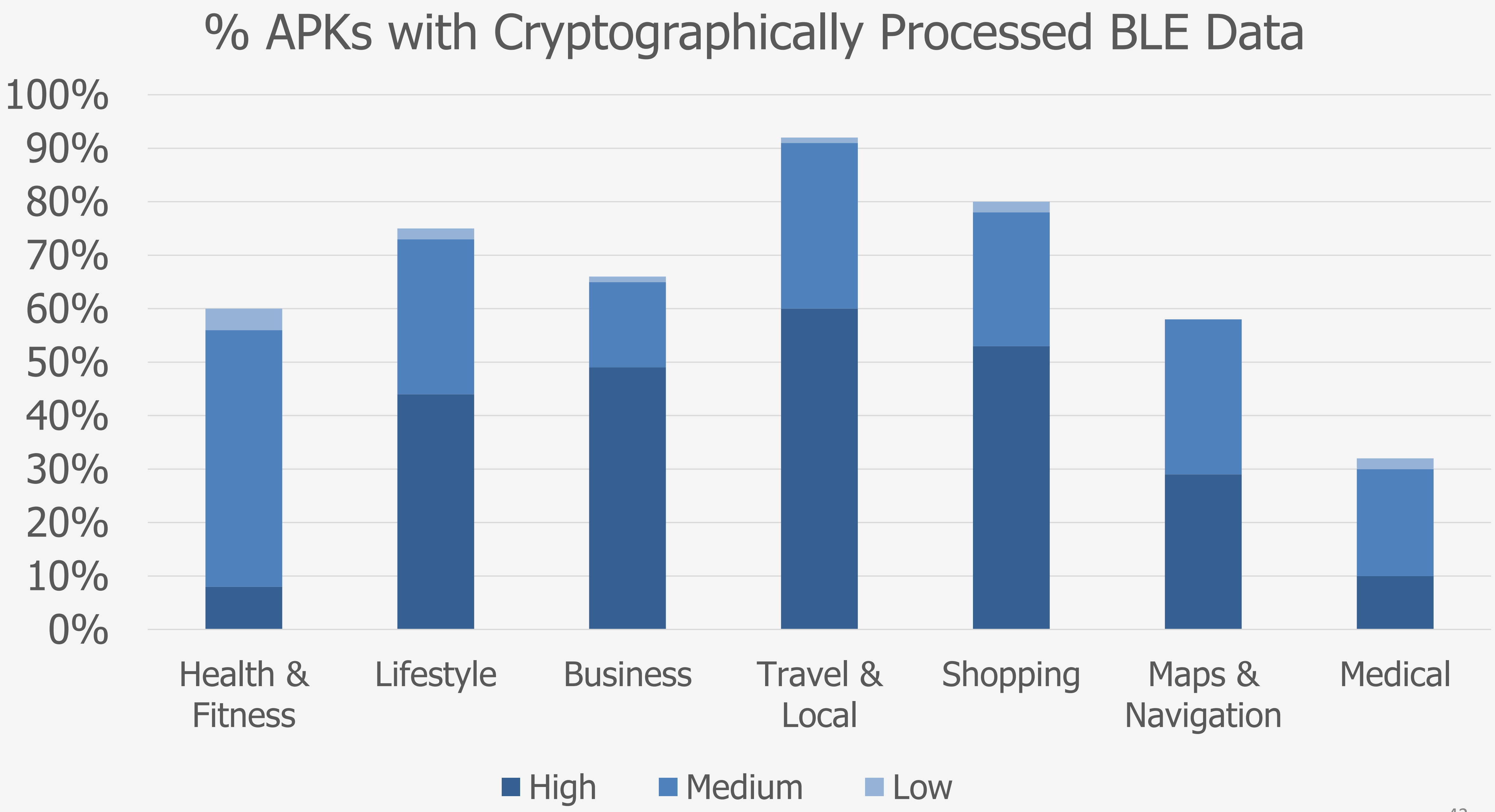
Access	Tool	Conf. Level	App Set	Detected	TP	FP	TN	FN	Precision	Recall	F-Measure
Read	Aman-droid	N/A	92	49	44	5	10	33	90%	57%	70%
	BLE Crypt-racer	High	92	62	58	4	11	19	94%	75%	83%
		Med	30	11	7	4	7	12	64%	37%	47%
		Low	19	12	8	4	3	4	67%	67%	67%
Write	Aman-droid	N/A	92	56	49	7	8	28	88%	64%	74%
	BLE Crypt-racer	High	92	50	46	4	11	31	92%	60%	72%
		Med	42	22	19	3	8	12	86%	61%	72%
		Low	20	10	5	5	3	7	50%	42%	45%

- Real-world APKs
  - Executed against 18,929 APKs (from Androzoo) that have BLE data access calls.

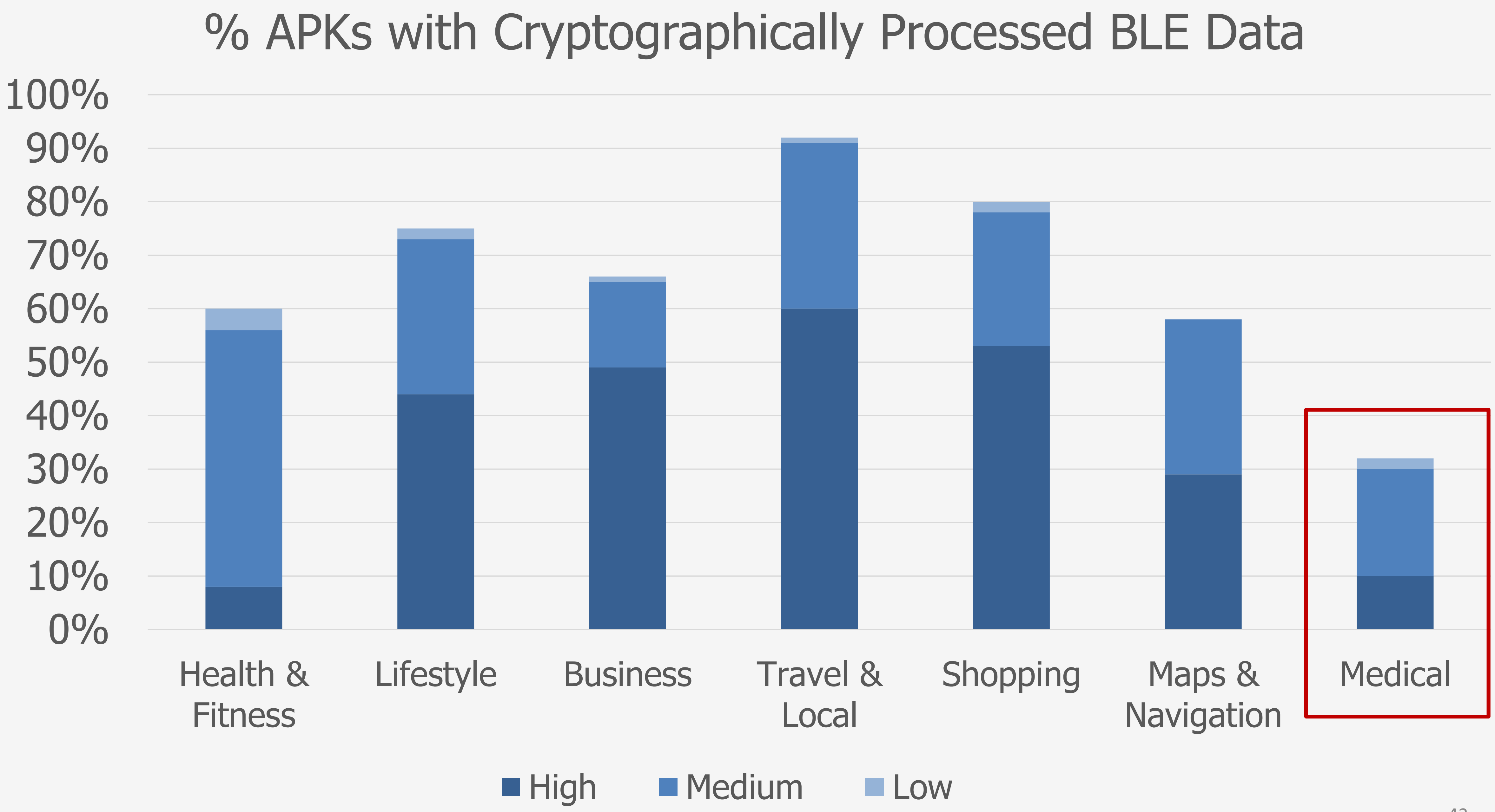
BLECryptracer Results



- Several APKs implement BLE functionality via 3rd party libraries.
  - Beacon, DFU, BLE “helper”/wrappers…
  - BLE writes: 63% APKs solely use libraries.
  - BLE reads: 58% use only libraries.
- App-specific BLE data access methods less likely to incorporate crypto.



% APKs with Cryptographically Processed BLE Data
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Health & Fitness
Lifestyle
Business
Travel & Local
Shopping
Maps & Navigation
Medical
High
Medium
Low


% APKs with Cryptographically Processed BLE Data
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Health & Fitness
Lifestyle
Business
Travel & Local
Shopping
Maps & Navigation
Medical
High
Medium
Low

- Cryptographical correctness (CogniCrypt)
  - ECB or other bad mode
  - Hardcoded keys
  - Non-random IVs
  - Incomplete operations

In Summary...

- Pairing-protected attributes on the BLE device can be read and written by any application on the Android device.
- Regardless of pairing method.
- Opportunistic data access enables malicious apps to request fewer permissions than legitimate apps.

- Different stakeholders involved. Difficult to determine responsibility.
- Currently, security is in the hands of developers.
- Almost half of all BLE APKs don’t protect BLE reads/writes. Also, bad crypto practices in some that do.
- 70% of “Medical” apps don’t protect BLE data.

https://github.com/projectbtle/BLECryptracer

Thank You