Rethinking the Security and Privacy of Bluetooth Low Energy

Zhiqiang Lin
Distinguished Professor of Engineering
zlin@cse.ohio-state.edu

09/30/2022
What is Bluetooth

Bluetooth wireless technology

- Low-cost, low-power
- Short-range radio
- For ad-hoc wireless communication
- For voice and data transmission
What is Bluetooth

- Apple AirTag
  - To be introduced in the first half of 2020

- Google Contact tracing

- Logitech keyboard

- Spark device
Why Named Bluetooth

Harald “Bluetooth” Gormsson

- King of Denmark 940-981.
- He was also known for his bad tooth, which had a very dark blue-grey shade.
- He united the Tribes of Denmark.

The Bluetooth wireless specification design was named after the king in 1997, based on an analogy that the technology would unite devices the way Harald Bluetooth united the tribes of Denmark into a single kingdom.
Why Named Bluetooth

Harald “Bluetooth” Gormsson

- King of Denmark 940-981.
- He was also known for his bad tooth, which had a very dark blue-grey shade.
- He united the Tribes of Denmark.

The Bluetooth wireless specification design was named after the king in 1997, based on an analogy that the technology would unite devices the way Harald Bluetooth united the tribes of Denmark into a single kingdom.
History of Bluetooth
Dr. Jaap Haartsen started a project named Bluetooth.
History of Bluetooth

1994: Bluetooth Prototype
1998: Bluetooth SIG
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth 1.0+1.0b
  - Many bugs
  - Bluetooth Basic Rate (~700KB/s)
- **1999**: Bluetooth SIG
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.0+1.0b
- **2001**: Bluetooth 1.1

- (~700KB/s) • Fixed security issues.
- (~700KB/s) • First marketable product version.
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.0+1.0b
- **2001**: Bluetooth 1.1 (~700KB/s)
- **2004**: Bluetooth 2.0/2.1 +EDR (~2.1MB/s)
- **2009**: Bluetooth 2.0/2.1 (~700KB/s)

- Introduced EDR (Enhanced Data Rate)
- Secure simple pairing (SSP) is introduced
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.0+1.0b
- **2001**: Bluetooth 2.0/2.1 +EDR (~700KB/s)
- **2004**: Bluetooth 2.0/2.1 +EDR (~2.1MB/s)
- **2009**: Introduced High-speed channel (~24MB/s)

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.1 (~700KB/s)
- **2009**: Bluetooth 3.0+HS (~24MB/s)
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.0+1.0b
- **2001**: Bluetooth 1.1 (~700KB/s)
- **2004**: Bluetooth 2.0/2.1 +EDR (~2.1MB/s)
- **2009**: Bluetooth 3.0+HS (~24MB/s)
- **2010**: Bluetooth 4.0 (~24MB/s)

- **•** Low energy (LE) protocol for IoT;
- **•** 128-bit encryption/LE Privacy and Whitelisting
History of Bluetooth

1994
- Bluetooth Prototype

1998
- Bluetooth SIG

1999
- Bluetooth 1.0+1.0b
- (~700KB/s)

2001
- Bluetooth 1.1
- (~700KB/s)

2004
- Bluetooth 2.0/2.1 +EDR
- (~2.1MB/s)

2009
- Bluetooth 3.0+HS
- (~24MB/s)

2010
- Bluetooth 4.0
- (~24MB/s)

2013
- Bluetooth 4.1
- (~700KB/s)

- Low energy (LE) protocol for IoT;
- 128-bit encryption/LE Privacy and Whitelisting
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.0+1.0b
- **2001**: Bluetooth 1.1 (~700KB/s)
- **2009**: Bluetooth 2.0/2.1 +EDR (~2.1MB/s)
- **2010**: Bluetooth 3.0+HS (~24MB/s)
- **2013**: Bluetooth 4.1 (~24MB/s)
- **2014**: Bluetooth 4.2

**Low energy (LE) protocol for IoT;**
**128-bit encryption/LE Privacy and Whitelisting**
History of Bluetooth

1994: Bluetooth Prototype
1998: Bluetooth SIG
1999: Bluetooth 1.0+1.0b
2001: Bluetooth 1.1 (~700KB/s)
2004: Bluetooth 2.0/2.1 +EDR (~2.1MB/s)
2009: Bluetooth 3.0+HS (~24MB/s)
2010: Bluetooth 4.0 (~24MB/s)
2013: Bluetooth 4.1
2014: Bluetooth 4.2
2016: Bluetooth 5.0 (~50MB/s)

Significant increase in range and data transfer rate

Introduction
BLE Security
BLE Privacy
Takeaway
History of Bluetooth
History of Bluetooth

- **1994**: Bluetooth Prototype
- **1998**: Bluetooth SIG
- **1999**: Bluetooth 1.0+1.0b
- **2001**: Bluetooth 1.1 (~700KB/s)
- **2004**: Bluetooth 2.0/2.1 +EDR (~2.1MB/s)
- **2009**: Bluetooth 3.0+HS (~24MB/s)
- **2010**: Bluetooth 4.0 (~24MB/s)
- **2013**: Bluetooth 4.1
- **2014**: Bluetooth 4.2
- **2016**: Bluetooth 5.0 (~50MB/s)
- **2019**: Bluetooth 5.1/5.2 (~50MB/s)
Total Annual Bluetooth Device Shipments

6.2 billion
annual shipments

8% CAGR
2019 - 2024

5 / 32
Total Annual Bluetooth Device Shipments

Annual Bluetooth Audio Streaming Device Shipments

1.54 billion annual shipments

7% CAGR

[Bar chart showing annual shipments from 2015 to 2022 with a 7% CAGR from 2015 to 2022.]
Total Annual Bluetooth Device Shipments
Total Annual Bluetooth Device Shipments

Annual Bluetooth Audio Streaming Device Shipments

Annual Bluetooth Phone, Tablet & PC Shipments

Annual Bluetooth Entertainments Shipments
Total Annual Bluetooth Device Shipments

Annual Bluetooth Audio Streaming Device Shipments

Annual Bluetooth Phone, Tablet & PC Shipments

Annual Bluetooth Entertainments Shipments

Annual Bluetooth Location Service Device Shipments

Apple AirTag
To be introduced in the first half of 2020
The General Workflow of Device Communication in TCP/IP Setting
The General Workflow of Device Communication in TCP/IP Setting

1. Listen to port 443
The General Workflow of Device Communication in TCP/IP Setting

1. Listen to port 443

2. <Request, 192.168.1.1, port 443>
The General Workflow of Device Communication in TCP/IP Setting

1. Listen to port 443
2. <Request, 192.168.1.1, port 443>
3. Connect
The General Workflow of Device Communication in TCP/IP Setting

1. Listen to port 443
2. <Request, 192.168.1.1, port 443>
3. Connect
4. Authentication (Transport Layer Security / Secure Sockets Layer)
The General Workflow of Device Communication in TCP/IP Setting

1. Listen to port 443
2. <Request, 192.168.1.1, port 443>
3. Connect
4. Authentication (Transport Layer Security / Secure Sockets Layer)
5. Communication
The General Workflow of Device Communication in TCP/IP Setting
The General Workflow of BLE IoT Devices and Companion Apps
The General Workflow of BLE IoT Devices and Companion Apps
The General Workflow of BLE IoT Devices and Companion Apps

1. Request for scan
2. Notify
The General Workflow of BLE IoT Devices and Companion Apps

1. Request for scan
2. Notify
3. Identify target device
The General Workflow of BLE IoT Devices and Companion Apps

1. Request for scan
2. Notify
3. Identify target device
4. Connect
The General Workflow of BLE IoT Devices and Companion Apps
The General Workflow of BLE IoT Devices and Companion Apps

1. Request for scan
2. Notify
3. Identify target device
4. Connect
5. Start pairing
6. Pairing

THE OHIO STATE UNIVERSITY
The General Workflow of BLE IoT Devices and Companion Apps

1. Request for scan
2. Notify
3. Identify target device
4. Connect
5. Start pairing
6. Pairing
7. Communication
Pairing Workflow

- Device
- OS
- App
Pairing Workflow

1. Start pairing
Pairing Workflow

1. Start pairing
2. Pairing feature exchange
**Pairing Workflow**

1. Start pairing
2. Pairing feature exchange
3. Authentication and encryption

**Pairing Methods**
- Just Works
- Passkey Entry
- Out of band
- Numeric Comparison
Pairing Workflow

1. Start pairing
   - LTK

2. Pairing feature exchange
   - Pairing method¹
   - LTK

3. Authentication and encryption
   - LTK

4. Key distribution (e.g. IRK)

Pairing Methods
- Just Works
- Passkey Entry
- Out of band
- Numeric Comparison
Pairing Workflow

1. Start pairing
2. Pairing feature exchange
3. Authentication and encryption
4. Key distribution (e.g. IRK)
5. Encrypted communication

Pairing Methods
- Just Works
- Passkey Entry
- Out of band
- Numeric Comparison
Workflow of Passkey Entry

Device A

Device B
Workflow of Passkey Entry
Workflow of Passkey Entry

Computer Security Laboratory
Device A
123456
PKA

Device B
PKB
Workflow of Passkey Entry

Device A

123456

PKA

Device B

123456

PKB
Workflow of Passkey Entry

Device A
123456
PKA
PKB
F(PKA,PKB,123456)

Device B
123456
PKA
PKB
F(PKA,PKB,123456)
Workflow of Passkey Entry

Device A

123456

PKA

PKM

Device B

PKB

PKM

Malicious Device

PKA, PKB, PKM, ?

K1 = F(PKA, PKM, 28643)

PKA

PKB

123456

K2 = F(PKB, PKM, 28643)
Workflow of Numeric Comparison

Device A

Device B

1. Device A and Device B initiate Bluetooth pairing request.
2. Both devices display a numeric code (123456).
3. User verifies code on both devices and confirms pairing.

Takeaway: Ensure both devices match numeric codes to establish secure pairing.
Workflow of Numeric Comparison

Device A

PKA

Device B

PKB

123456

Cancel Pair

123456

Cancel Pair
Workflow of Numeric Comparison

Device A

PKA

PKB

Hash(PKA,PKB)

123456

Device B

PKA

PKB

Hash(PKA,PKB)

123456

Hash(PKA,PKB)

123456
Workflow of Numeric Comparison

Device A
- PKA
- Hash(PKA, PKB)
- F(PKA, PKB)
- 123456

Device B
- PKB
- Hash(PKA, PKB)
- F(PKA, PKB)
- 123456
Workflow of Numeric Comparison

Device A
PKA
PKM
Hash(PKA,PKM)
123456
K=F(PKA,PKM)

Malicious Device
PKM
PKB
Hash(PKB,PKM)

Device B
PKM
PKB
Hash(PKB,PKM)
654321
K=F(PKB,PKM)

K=F(PKB,PKM)

Workflow of Out of Band

Device A

Device B
Workflow of Out of Band
Workflow of Out of Band
Workflow of Out of Band

Device A

PKA

PKB

Device B

K = F(PKA, PKM, S)

K = F(PKB, PKM, S)

Out-of-Band Secure Channel (S)
Workflow of Out of Band

Device A → Malicious Device → Device B

PKA, PKM, PKB, PKM, ?

K1 = F(PKA, PKM, S)

Out-of-Band Secure Channel (S)

K2 = F(PKB, PKM, S)
Workflow of Justworks

Device A

Device B
Workflow of Justworks
Workflow of Justworks

Device A

PKA

PKB

Device B

K=\text{F}(\text{PKA}, \text{PKM}, 00000)

K=\text{F}(\text{PKB}, \text{PKM}, 000000)
Workflow of Justworks

Device A
PKA
PKM
K1 = F(PKA, PKM, 00000)

Malicious Device
PKB
PKM
F(PKB, PKM, 00000)

Device B
PKM
PKB
K2 = F(PKB, PKM, 00000)
Bluetooth Security Levels and SCO mode

1. **Level 1:** Plaintext (None)
2. **Level 2:** Encrypted (Just Works)
3. **Level 3:** Authenticated (Passkey, Entry, Numeric Comparison, and Out of Band)
Bluetooth Security Levels and SCO mode

1. **Level 1**: Plaintext (None)
2. **Level 2**: Encrypted (Just Works)
3. **Level 3**: Authenticated (Passkey, Entry, Numeric Comparison, and Out of Band)
4. **Secure Connection Only (SCO) mode**: Enforced Passkey, Numeric Comparison, and Out of Band

**Our Observation**

Although BLE specifies the SCO mode for a peripheral that provides services, it does not explicitly define (or require) the SCO mode for a master, which is also the pairing initiator.
Root Cause

1. Start pairing

2. Pairing feature exchange

3. Authentication and encryption
   
   **Just Works**

4. Disconnect

4. Communication
Root Cause

1. Start pairing
2. Pairing feature exchange
3. Authentication and encryption

Just Works

4. Disconnect
4. Communication

Vulnerable

Device
OS
App
Downgrade Attacks

Paired with a secure pairing method (Passkey Entry/Numeric Comparison)
Downgrade Attacks

1. Impersonate the victim device and deploy attacks against the mobile

Paired with a secure pairing method (Passkey Entry/Numeric Comparison)
Downgrade Attacks

1. Impersonate the victim device and deploy attacks against the mobile
2. Use the stolen information (i.e., IRK) to create a Fake mobile

Paired with a secure pairing method (Passkey Entry/Numeric Comparison)
Downgrade Attacks

1. Impersonate the victim device and deploy attacks against the mobile.
2. Use the stolen information (i.e., IRK) to create a Fake mobile.

Paired with a secure pairing method (Passkey Entry/Numeric Comparison)
Downgrade Attacks

1. Impersonate the victim device and deploy attacks against the mobile.
2. Use the stolen information (i.e., IRK) to create a Fake mobile.
3. Deploy attacks against the device.

Paired with a secure pairing method (Passkey Entry/Numeric Comparison)
Downgrade Attacks

The Tested BLE devices

MITM attack against BLE keyboards

CVE-2020-9770
Downgrade Attacks

1. Impersonate the victim device and deploy attacks against the mobile.
2. Use the stolen information (i.e., IRK) to create a fake mobile.
3. Deploy attacks against the device.

Paired with a secure pairing method (Passkey Entry/Numeric Comparison).

The Format of A Bluetooth Packet

- **Preamble (1 byte)**
- **Access Address (4 bytes)**
- **Packet Data Unit (2 - 257 bytes)**
- **CRC (3 bytes)**
The Format of A Bluetooth Packet
The Format of A Bluetooth Packet
The Format of A Bluetooth Packet
The Format of A Bluetooth Packet
Bluetooth Sniffers

Ubertooth One Sniffer
125 USD

Adafruit LE sniffer
25 USD
Bluetooth Sniffers

Alice's phone

Bob's phone

T1: 52:09:4A:87:0A:A1

T2: 52:09:4A:87:0A:A1
Bluetooth Address Types

- Bluetooth Address
- Random Address
- Public Address
Bluetooth Address Types

- Bluetooth Address
  - Random Address
  - Public Address
    - Random Private Address
    - Random Static Address
Bluetooth Address Types

- Bluetooth Address
  - Random Address
  - Public Address
  - Random Private Address
  - Random Static Address
    - Random Private Resolvable Address
    - Random Private Non-Resolvable Address
How to Avoid Being Tracked: MAC Address Randomization

Identity Resolving Key (IRK)
How to Avoid Being Tracked: MAC Address Randomization

Identity Resolving Key (IRK)

(I) RPA Generation
How to Avoid Being Tracked: MAC Address Randomization

Identity Resolving Key (IRK)

Resolvable Private Address (RPA)

\[
\text{rand} = \text{prand} + \text{type} \\
\text{hash} = H(\text{IRK}, \text{rand})
\]

<table>
<thead>
<tr>
<th>Type</th>
<th>prand</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 (2bits)</td>
<td>0x00...3 (22bits)</td>
<td>0x00...04 (24bits)</td>
</tr>
</tbody>
</table>
How to Avoid Being Tracked: MAC Address Randomization

Identity Resolving Key (IRK)

(I) RPA Generation

Resolvable Private Address (RPA)

(II) RPA Resolution

<table>
<thead>
<tr>
<th>Type</th>
<th>prand</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 (2bits)</td>
<td>0x00...3 (22bits)</td>
<td>0x00...04 (24bits)</td>
</tr>
</tbody>
</table>

$$\text{hash} = H(\text{IRK}, \text{rand})$$

$$\text{rand} = \text{prand} + \text{type}$$
How to Avoid Being Tracked: MAC Address Randomization

I. Advertising filter policy

Only allows its allowlisted central to connect (e.g., Smart Earbuds)

II. Advertising filter policy

Automatically connects its allowlisted peripherals (e.g., Smart Earbuds)

Allowlist creates a one-to-one /one to multiple relationship between devices
Allowlist

Ubertooth One Sniffer
125 USD

Adafruit LE sniffer
25 USD
Our Insights (I) — Allowlist Side Channel

<table>
<thead>
<tr>
<th>NO.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:04</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>ADV_IND</td>
</tr>
</tbody>
</table>
Our Insights (I) — Allowlist Side Channel

<table>
<thead>
<tr>
<th>NO.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:04</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>ADV_IND</td>
</tr>
<tr>
<td>2</td>
<td>00:00:08</td>
<td>7e:D7:8E:C7:8e:51</td>
<td>58:D7:8E:C7:8e:31</td>
<td>SCAN_REQ</td>
</tr>
<tr>
<td>3</td>
<td>00:00:12</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>SCAN_RSP</td>
</tr>
</tbody>
</table>
Our Insights (I) — Allowlist Side Channel

<table>
<thead>
<tr>
<th>NO.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:04</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>ADV_IND</td>
</tr>
<tr>
<td>2</td>
<td>00:00:08</td>
<td>7e:D7:8E:C7:8e:51</td>
<td>58:D7:8E:C7:8e:31</td>
<td>SCAN_REQ</td>
</tr>
<tr>
<td>3</td>
<td>00:00:12</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>SCAN_RSP</td>
</tr>
<tr>
<td>4</td>
<td>00:00:16</td>
<td>4f:b7:8E:C7:8e:38</td>
<td>58:D7:8E:C7:8e:31</td>
<td>SCAN_REQ</td>
</tr>
<tr>
<td>5</td>
<td>00:00:24</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>ADV_IND</td>
</tr>
</tbody>
</table>
Our Insights (I) — Allowlist Side Channel

<table>
<thead>
<tr>
<th>NO.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:04</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>ADV_IND</td>
</tr>
<tr>
<td>2</td>
<td>00:00:08</td>
<td>7e:D7:8E:C7:8e:51</td>
<td>58:D7:8E:C7:8e:31</td>
<td>SCAN_REQ</td>
</tr>
<tr>
<td>3</td>
<td>00:00:12</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>SCAN_RSP</td>
</tr>
<tr>
<td>4</td>
<td>00:00:16</td>
<td>4f:b7:8E:C7:8e:38</td>
<td>58:D7:8E:C7:8e:31</td>
<td>SCAN_REQ</td>
</tr>
<tr>
<td>5</td>
<td>00:00:24</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>ADV_IND</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>00:15:08</td>
<td>73:D7:8E:C7:8e:45</td>
<td>58:D7:8E:C7:8e:31</td>
<td>SCAN_REQ</td>
</tr>
<tr>
<td>201</td>
<td>00:15:12</td>
<td>58:D7:8E:C7:8e:31</td>
<td>Broadcast</td>
<td>SCAN_RSP</td>
</tr>
</tbody>
</table>
Our Insights (I) — Allowlist Side Channel
Our Insights (I) — Allowlist Side Channel
Our Insights (I) — Allowlist Side Channel
Our Insights (I) — Allowlist Side Channel
Our Insights (I) — Allowlist Side Channel
Passive Bluetooth Address Tracking (BAT) Attacks
Passive Bluetooth Address Tracking (BAT) Attacks
Passive Bluetooth Address Tracking (BAT) Attacks
Passive Bluetooth Address Tracking (BAT) Attacks
Passive Bluetooth Address Tracking (BAT) Attacks

Diagram:
- Mobile A
- Device P
- Mobile B
- RPA-P1
- RPA-B1
- SCAN_REQ
- SCAN_RSP
- Broadcast
- T0
Passive Bluetooth Address Tracking (BAT) Attacks
Passive Bluetooth Address Tracking (BAT) Attacks
Passive Bluetooth Address Tracking (BAT) Attacks

Introduction

BLE Security

BLE Privacy

Takeaway

Passive Bluetooth Address Tracking (BAT) Attacks

Mobile A

Device P

Mobile B

RPA-B1

RPA-B2

RPA-P1

RPA-P2

SCAN_REQ

SCAN_RSP

Broadcast

T0

T1
Passive Bluetooth Address Tracking (BAT) Attacks
Our Insights (II) — MAC Address Replay
Our Insights (II) — MAC Address Replay

Identity Resolving Key (IRK)

(I) RPA Generation

Identity Resolving Key (IRK)
Our Insights (II) — MAC Address Replay

Identity Resolving Key (IRK)

(I) RPA Generation

Resolvable Private Address (RPA)

hash = H(IRK, rand)

rand = prand + type

(II) RPA Resolution

<table>
<thead>
<tr>
<th>Type</th>
<th>prand</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 (2bits)</td>
<td>0x00...3</td>
<td>0x00...04</td>
</tr>
<tr>
<td></td>
<td>(22bits)</td>
<td>(24bits)</td>
</tr>
</tbody>
</table>

RPA1
Our Insights (II) — MAC Address Replay

Identity Resolving Key (IRK)

(I) RPA Generation

rand = prand + type

hash = H(IRK, rand)

Resolvable Private Address (RPA)

(II) RPA Resolution

<table>
<thead>
<tr>
<th>Type</th>
<th>prand</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>0x00…3</td>
<td>0x00…04</td>
</tr>
</tbody>
</table>

Type

prand

Hash
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks
Active Bluetooth Address Tracking (BAT) Attacks

SABLE — Passive Defense
SABLE — Passive Defense

Device P 
Mobile B 
Broadcast 
Mobile A 
SCAN_REQ 
SCAN_REQ 
T0 RPA-B1 
RPA-P1
SABLE — Passive Defense
SABLE — Passive Defense
SABLE — Passive Defense
SABLE — Active Defense

Mobile A

Device P

Broadcast

SCAN_REQ

Mobile B

RPA-P1

SCAN_REQ

SCAN_RSP

RPA-P2

SCAN_REQ

SCAN_RSP

RPA-P3

T0

T1

RPA-B1

SCAN_REQ

SCAN_RSP

RPA-B2

SCAN_REQ

SCAN_RSP

RPA-B3

T2

T1 - T0 != T2 - T1
SABLE — Active Defense

Identity Resolving Key (IRK)
Sequence Number Tc
SABLE — Active Defense

Identity Resolving Key (IRK)
Sequence Number Tc

Resolvable Private Address (RPA)

\[ \text{hash} = H(\text{IRK}, \text{rand}, \text{Tc}) \]

(I) RPA Generation

<table>
<thead>
<tr>
<th>Type</th>
<th>prand</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>0x00...3</td>
<td>0x00...04</td>
</tr>
</tbody>
</table>
SABLE — Active Defense

Identity Resolving Key (IRK)  
Sequence Number Tc  

Resolvable Private Address (RPA)

\[
\text{hash} = H(\text{IRK}, \text{rand}, Tc)
\]

\[
\text{rand} = \text{prand} + \text{type}
\]

<table>
<thead>
<tr>
<th>Type</th>
<th>prand</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>0x00...3</td>
<td>0x00...04</td>
</tr>
</tbody>
</table>
Performance of SABLE

- Battery %:

  - Red: w/o SABLE
  - Green: w/ SABLE

  - Graph shows a substantial increase in battery life with SABLE.

- Time (minutes):

  - Central:
    - w/o SABLE: 67.95
    - w/ SABLE: 88.49

  - Peripheral:
    - w/o SABLE: 30.88
    - w/ SABLE: 94.46

These results indicate significant performance improvements with SABLE.
Lesson Learned (1/3): Bluetooth Communication Can Be Downgraded

- Bluetooth protocol can be downgraded
- There are many stages that are not part of the pairing protocol, but they are, in fact, closely related to pairing security.
- A systematic analysis of different stages (other than pairing), including the error handling of BLE communication, is needed.
Lesson Learned (2/3): New Features Need More Investigation

- BLE introduces multiple new features (e.g., Allow-list)
- Those new features need to be understood and investigated. For example, Cross-transport key derivation (CTKD); Authorization; The Connection Signature Resolving Key (CSRK).
Lesson Learned (3/3): Formal Method Can Help Improve BLE Security

- The specification is often confusing and not consistent across chapters.
- The confusion may lead to the fact that different vendors implement BLE protocols in quite different ways, for example, for error handling, and IRK use.
- Converting the Bluetooth specification to formal model (e.g., using NLP), and formally verify the entire protocol would help.
- See our NDSS’23 paper.