
black hat®
ASIA 2026
APRIL 21-24, 2026
MARINA BAY SANDS / SINGAPORE

WhisperPair: A Security Analysis of Google Fast Pair

Sayon Duttagupta*, **Nikola Antonijević**, Bart Preneel (COSIC, KU Leuven)

Seppe Wyns*, Dave Singelée (DistriNet Group T, KU Leuven)

ANDY GREENBERG | LILY HAY NEWMAN | SECURITY

JAN 15, 2026 7:00 AM

Hundreds of Millions of Audio Devices Need a Patch to Prevent Wireless Hacking and Tracking

Flaws in how 17 models of headphones and speakers use Google's one-tap Fast Pair Bluetooth protocol have left devices open to eavesdroppers and stalkers.

About us
Speakers

Sayon Duttagupta

Research Scientist

COSIC, KU Leuven

Seppe Wyns

Doctoral Researcher

DistriNet (Group T), KU Leuven

Nikola Antonijević

Doctoral Researcher

COSIC, KU Leuven

Bluetooth

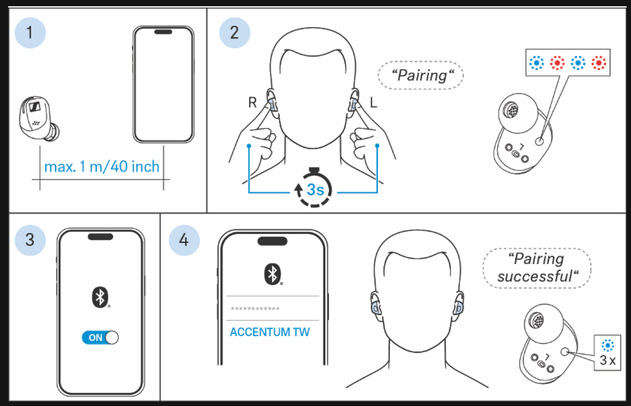
Pairing a new device

- Establishes **trust** between two devices
- Exchange keys to encrypt communication
- Pairing process can be **cumbersome**

Why Bluetooth remains an ‘unusually painful’ technology after two decades

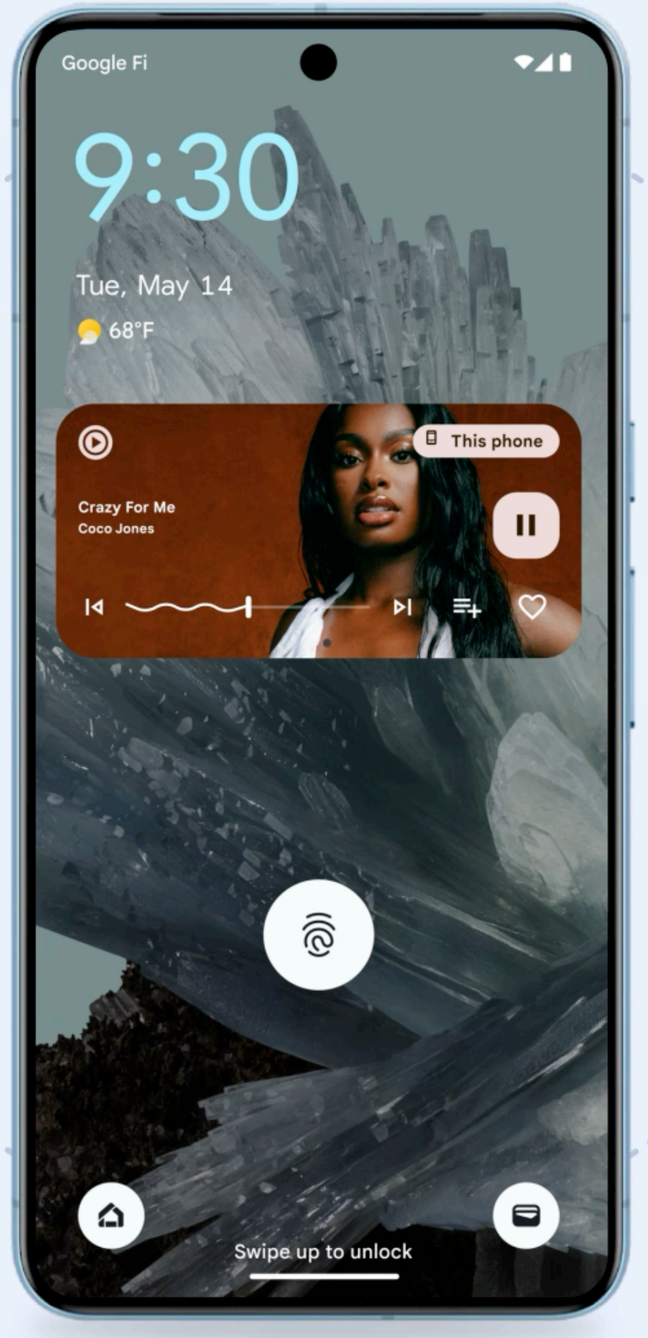
By Catherine Thorbecke, CNN Business

4 min read · Published 5:02 AM EDT, Sun July 10, 2022



Reference: https://firmware.s-consumer-cloud.com/help/products/atw1/en/manual/ATW1_Manual_EN/topic_ATW1_Manual_EN14.htm

Fast Pairing Protocols



Source: Google

Google Fast Pair

Motivation

- Improves the user experience of pairing Bluetooth accessories
  - **One-tap** pairing

- **Widespread** support: Sony, JBL, Bose,…
- Bluetooth security has been studied extensively, what about Fast Pair?

Google Fast Pair
Terminology

Accessory: **Provider**

Smartphone: **Seeker**

Fast Pair performs a Bluetooth *Classic* (BR/EDR) pairing using Bluetooth *Low Energy* (BLE)

Image Source: Google

Google Fast Pair

How does it work?

1. Provider advertises **Model ID** when in pairing mode

24-bit identifier assigned by Google

Provider

Google Fast Pair

How does it work?

2. Seeker fetches model **public key**

Provider

Seeker

Every model has the same hardcoded public/private key pair

Seeker can get the public key of the model using a Google API

Google Fast Pair

How does it work?

3. ECDH is used to establish an **encrypted** channel
4. Seeker sends an (encrypted) **pairing request** to the Provider

Google Fast Pair

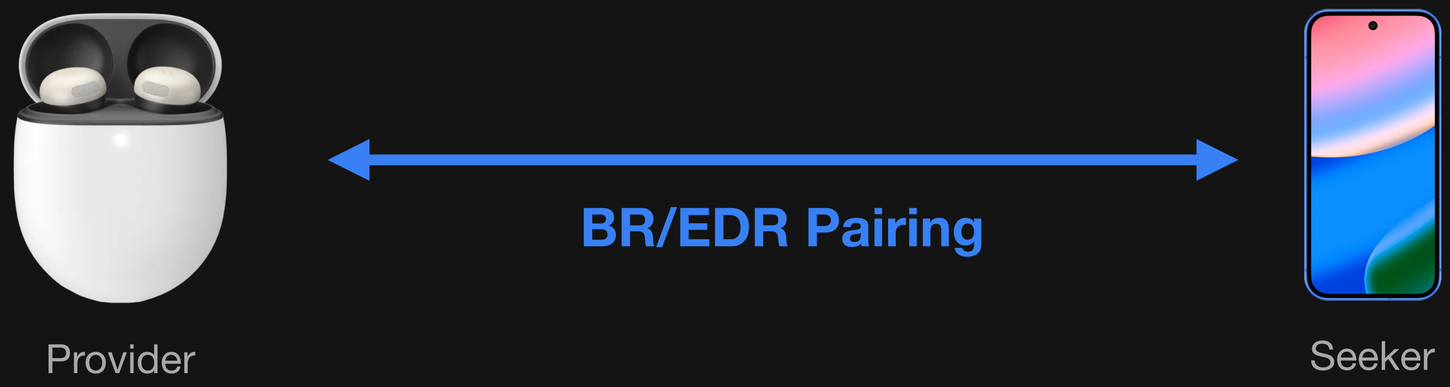
How does it work?

5. Provider replies with **Bluetooth Classic address**

Google Fast Pair

How does it work?

6. Seeker initiates **Bluetooth Classic pairing**

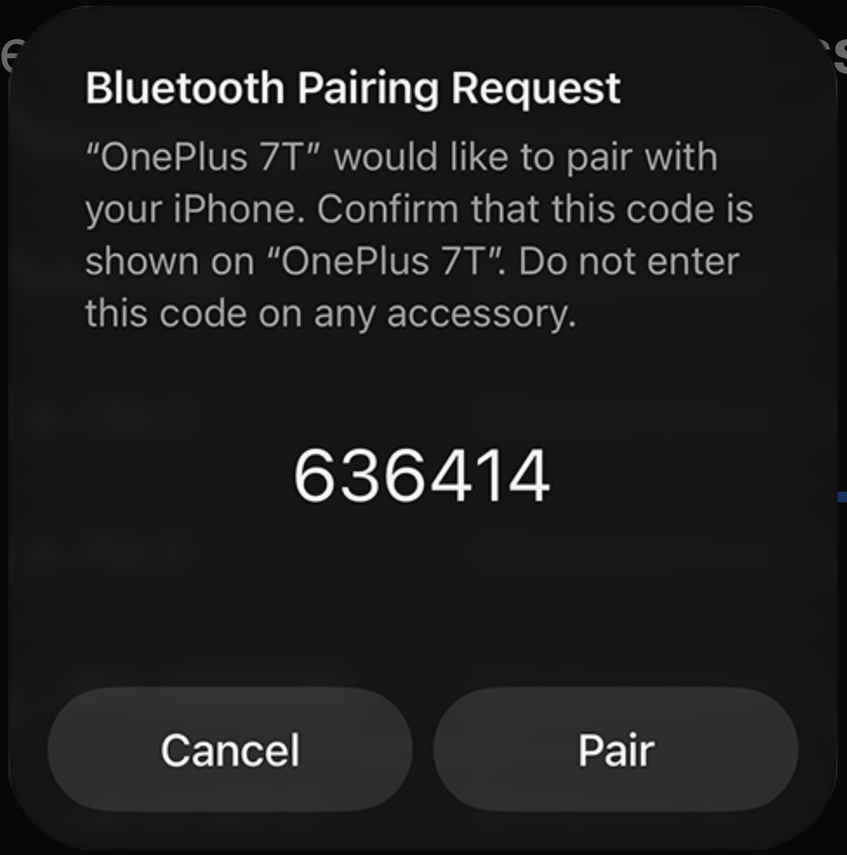


Google Fast Pair

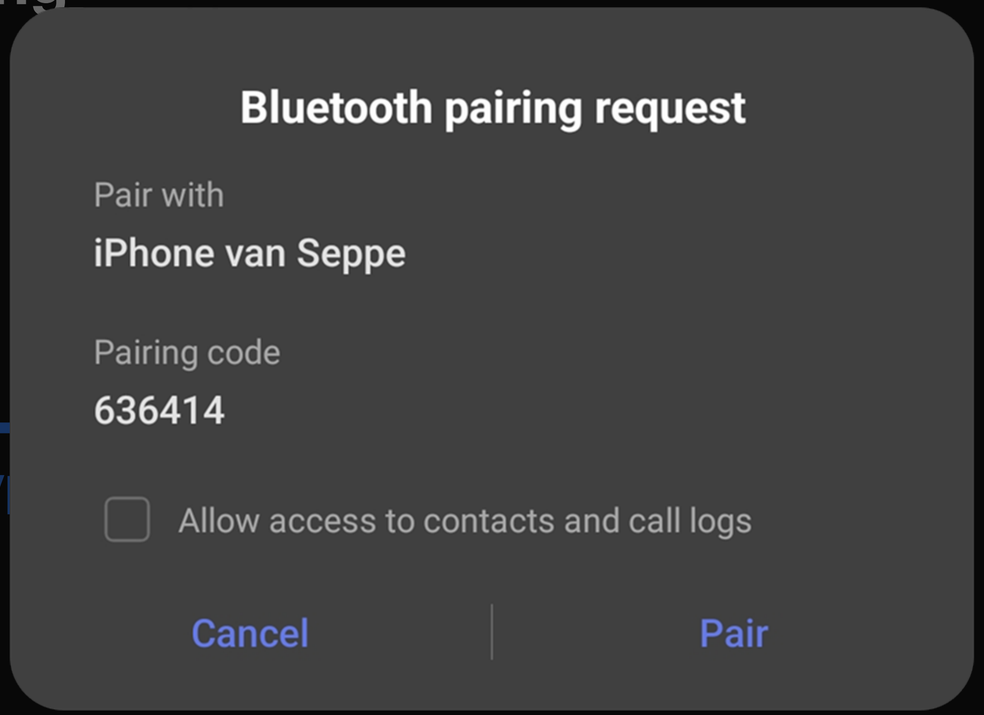
How does it work?

Under the hood:

6. See sic pairing



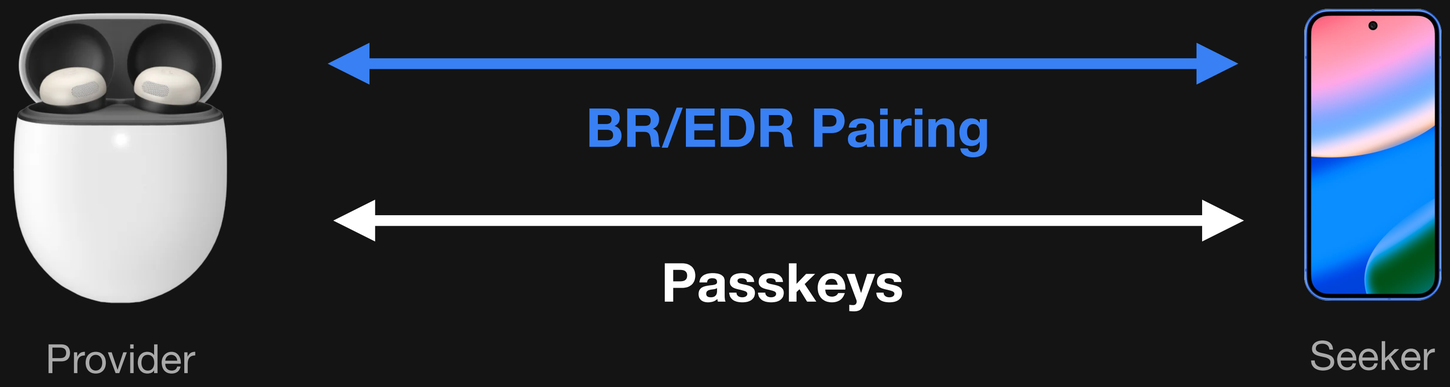
BR/



Google Fast Pair

How does it work?

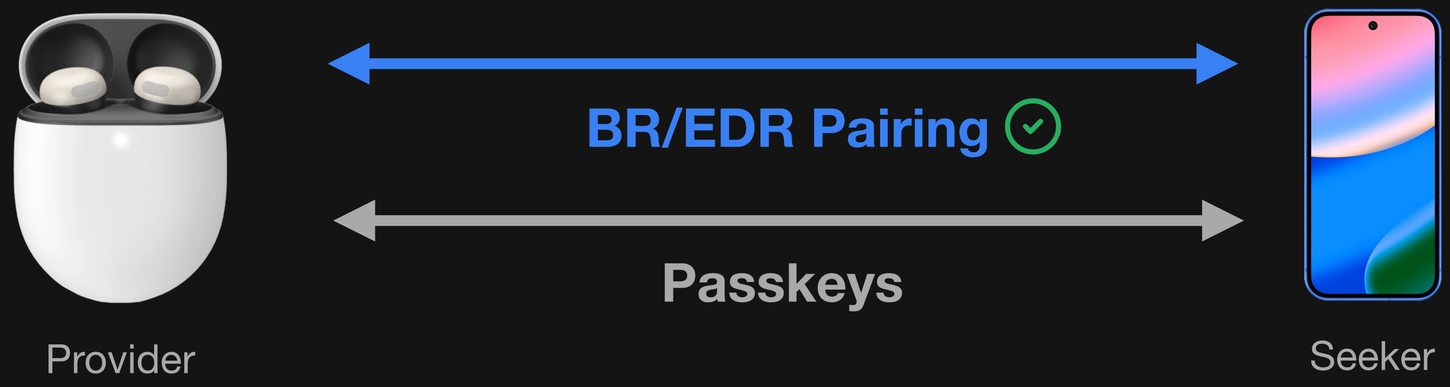
7. Pairing requires **passkey confirmation**
8. Passkeys are exchanged using BLE



Google Fast Pair

How does it work?

9. Bluetooth Classic Pairing is **confirmed**



Google Fast Pair

How does it work?

Provider

1. Pairing Request

2. BR/EDR Address

3. BR/EDR Pairing

4. Exchange Passkeys

Seeker

The Pairing State Predicate

Bluetooth Pairing Modes

Pairing Bluetooth devices

- When should a Bluetooth Classic device accept new connections?

→ **Pairing mode**

  - Often a physical trigger
  - Makes the device *discoverable* and *pairable*

⚠️ **Fast Pair enables pairing while the Provider is not in pairing mode**

Bluetooth Pairing Modes

Fast Pair

- Trusted devices can **initiate a pairing** using an account key a_k
- Seeker will encrypt its pairing request using a_k

Bluetooth Pairing Modes

Fast Pair

- Trusted devices can **initiate a pairing** using an account key a_k
- Seeker will encrypt its pairing request using a_k

- Consequences:
  1. Provider is **always** advertising
  2. The Fast Pair Service is **always** available

The Pairing State Predicate

What does the specification say?

When handling a write request from a Fast Pair Seeker, the Fast Pair Provider shall do the following:

1. If the optional Public Key field **is present**:
    a. If the device is not in pairing mode, ignore the write and exit.
    b. Otherwise:
        i. Use the received Public Key (a 64-byte point on the secp256r1 elliptic curve), the pre-installed Anti-Spoofing Private Key - also secp256r1, and the Elliptic-Curve Diffie-Hellman algorithm to generate a 256-bit AES key.
        ii. Use SHA-256 to hash the 256-bit AES key.

The Pairing State Predicate

What does the specification say?

When handling a write request from a Fast Pair Seeker, the Fast Pair Provider shall do the following:

1. If the optional Public Key field **is present**:

   a. If the device is not in pairing mode, ignore the write and exit.

   b. Otherwise:

A lot of manufacturers do not implement this check correctly!

The Pairing State Predicate

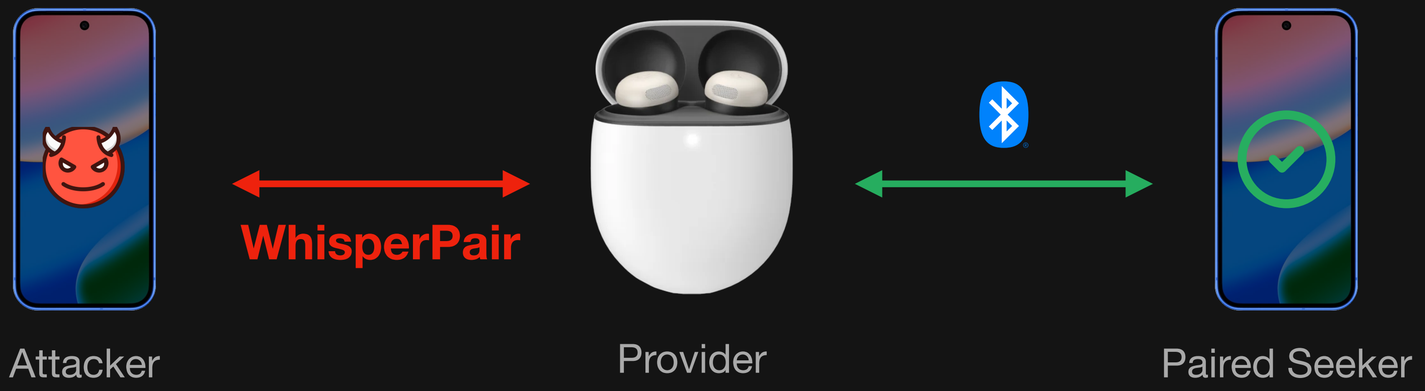
Impact

- A vulnerable provider allows **unauthorised** pairing
- Pairing without user intent

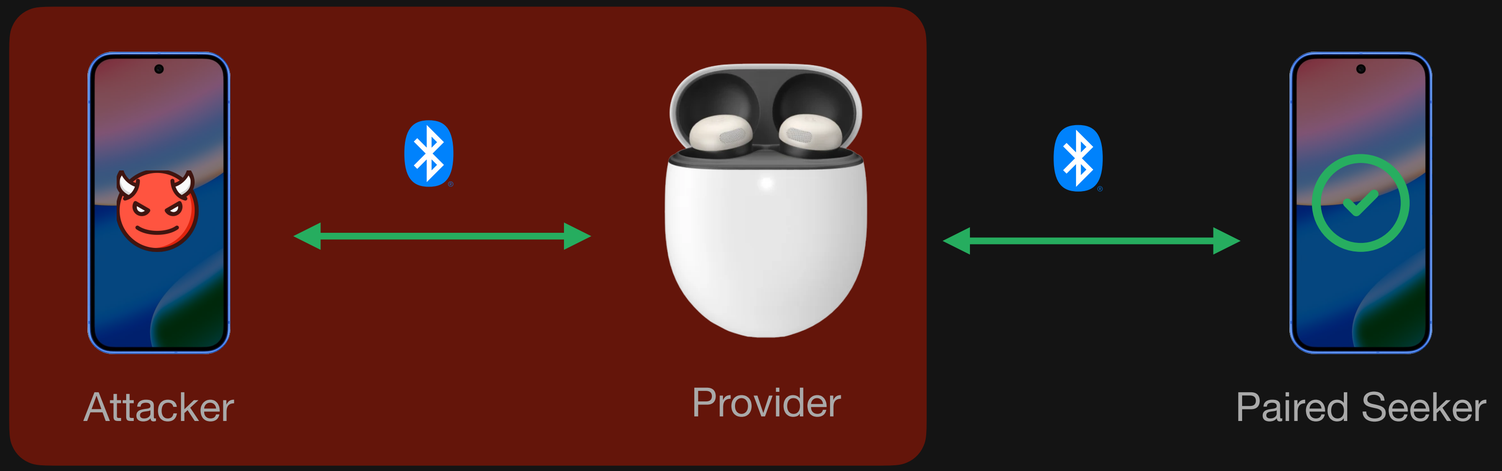
WhisperPair Walkthrough

Typical example

WhisperPair attack:



Device hijack:



From Flaw to Exploit

1. Finding a Fast Pair Provider

Advertising: When discoverable

When the Provider device is BR/EDR discoverable (that is, in pairing mode), it shall advertise Fast Pair Model ID Data over BLE, and the BLE address shall not be rotated.

Advertising: When not discoverable

When not discoverable (that is, not in pairing mode), the Provider device shall advertise Fast Pair Account Data, using the following guidelines.

Source: Fast Pair Specification

- Always advertising the Fast Pair UUID: `0xFE2C`

From Flaw to Exploit

2. Getting the public key of the Provider

```
const model = await getDeviceInfo(device.modelId);
```

- Intercepted traffic from Android device
- (Now removed) code from AOSP
- Protocol Buffer schemas

From Flaw to Exploit

3. Getting a pairing message

```
const model = await getDeviceInfo(device.modelId);

const { cipher, decipher, payload } = protocol.generateKeyBasedPairingMessage(
    device.address,
    model.publicKey,
);
```

From Flaw to Exploit

4. Writing the pairing message

```
const model = await getDeviceInfo(device.modelId);

const { cipher, decipher, payload } = protocol.generateKeyBasedPairingMessage(
    device.address,
    model.publicKey,
);

await writeToKeyBasedPairingCharacteristic(payload);
```

When the provider is not in pairing mode!

From Flaw to Exploit

5. Decoding the response

```
const onNotify = async (value: Buffer) => {




}
```

From Flaw to Exploit

5. Decoding the response

```
const onNotify = async (value: Buffer) => {

    const payload = decipher(value);









}
```

From Flaw to Exploit

5. Decoding the response

```
const onNotify = async (value: Buffer) => {
    const payload = decipher(value);
    const bluetoothClassicAddress = payload.slice(1, 7);




}
```

From Flaw to Exploit

6. Establishing a Bluetooth Classic pairing

```
const onNotify = async (value: Buffer) => {
    const payload = decipher(value);
    const bluetoothClassicAddress = payload.slice(1, 7);
    const session = new BluetoothAdapter();
    const passkey = await session.initiatePairing(bluetoothClassicAddress);




}
```

From Flaw to Exploit

7. Sending the passkey to the Provider

```
const onNotify = async (value: Buffer) => {
    const payload = decipher(value);
    const bluetoothClassicAddress = payload.slice(1, 7);
    const session = new BluetoothAdapter();
    const passkey = await session.initiatePairing(bluetoothClassicAddress);
    const reply = await protocol.generatePasskeyMessage(cipher, passkey);
    await writeToPasskeyCharacteristic(reply);


}
```

From Flaw to Exploit

8. Confirming the Bluetooth Classic pairing

```
const onNotify = async (value: Buffer) => {
    const payload = decipher(value);
    const bluetoothClassicAddress = payload.slice(1, 7);
    const session = new BluetoothAdapter();
    const passkey = await session.initiatePairing(bluetoothClassicAddress);
    const reply = await protocol.generatePasskeyMessage(cipher, passkey);
    await writeToPasskeyCharacteristic(reply);
    await session.confirm(signal);
}
```

Attack Implications

Impact
WhisperPair attack

- Attacker can initiate and complete the pairing procedure **without permission**

- Attacker has **complete** control
  - Play media
  - Control volume
  - Activate microphones
  - Device specific features (noise cancelling, device switching)

Practicality
WhisperPair attack

- Accessory needs to be turned on
- **No user interaction** required (zero-click)
- Succeeds within seconds (~10 s) at realistic distances (14 m)

Public Transport

Cafés, Restaurants

Parks, Public Spaces

Practicality

WhisperPair attack

- Accessory needs to be turned on
- **No user interaction** required (zero-click)
- Succeeds withi

Attacker just needs to be in Bluetooth range

Public Transport

Cafés, Restaurants

Parks, Public Spaces

Practicality

WhisperPair attack

- Fast Pair **cannot be disabled**
- Flaw in the firmware of the accessory→ non-Android users ( ) **also affected**
- Most devices will sound a brief audible cue
- Exact behaviour depends on the device

Attack is indistinguishable from the device disconnecting and reconnecting

Covert Location Tracking

But wait, there is more…

Android

About Features Devices Learn

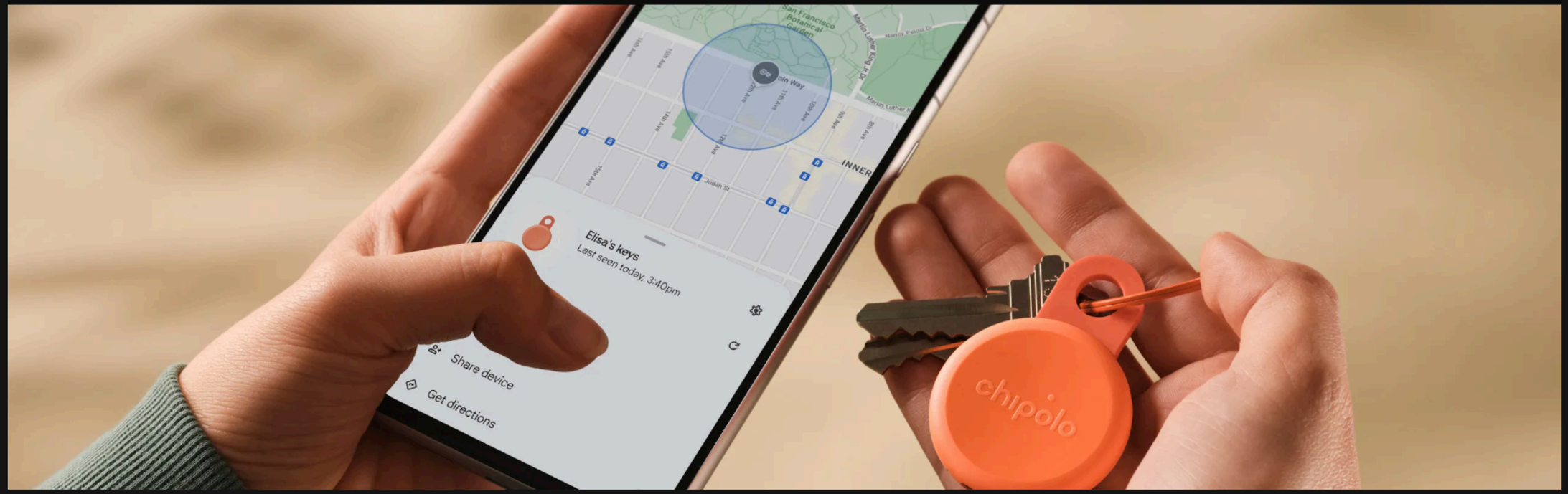
Shop phones

About Find Hub

Overview Compatible devices

Explore featured findable tags and devices.

From smart tracking tags, trackable earbuds, to headphones, know where your most important items are with Find Hub. Browse the full list of supported accessories and devices from top brands.



Find more than just phones.

From earbuds to luggage, Find Hub lets you locate all kinds of items in the same app. Look for the “Works with Android” badge to see compatible accessories.

Shop partners →

Wearables

Headphones

Tracking Tags

Find Hub

Overview

- **Extends** *Google Fast Pair* accessories with findable behaviour
- Crowdsourced **location** reporting via nearby  Android devices
- Account binding through an Owner Account Key (OAK)
    a. First account key becomes the **permanent** owner key
    b. Can only be removed by factory reset

Find Hub

Overview

- **Extends** *Google Fast Pair* accessories with findable behaviour
- Crowdsourced **location** reporting via nearby  Android devices
- Account binding through an Owner Account Key (OAK)
    a. First account key becomes the **permanent** owner key 🔑
    b. Can only be removed by factory reset 🔒

Devices that have never been connected to an Android device do *not* have an OAK

Find Hub

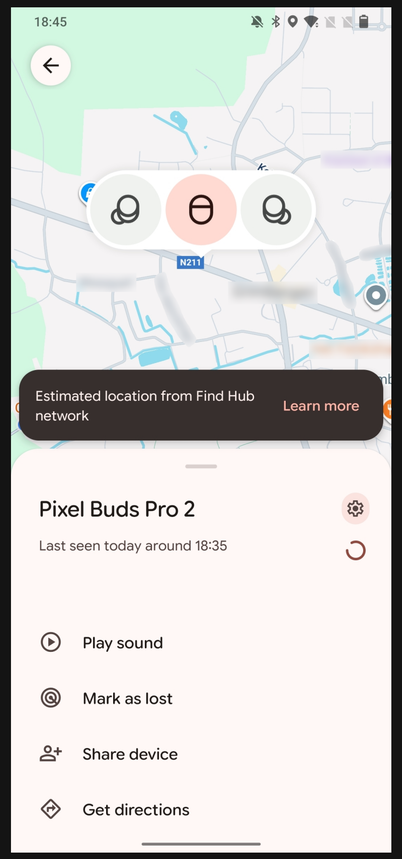
Covert Account Binding

- Attacker exploits missing pairing mode enforcement to set the OAK
- iOS or non-Android users that never enrolled in Find Hub
- Device becomes **bound to the attacker's Google account** 🔒

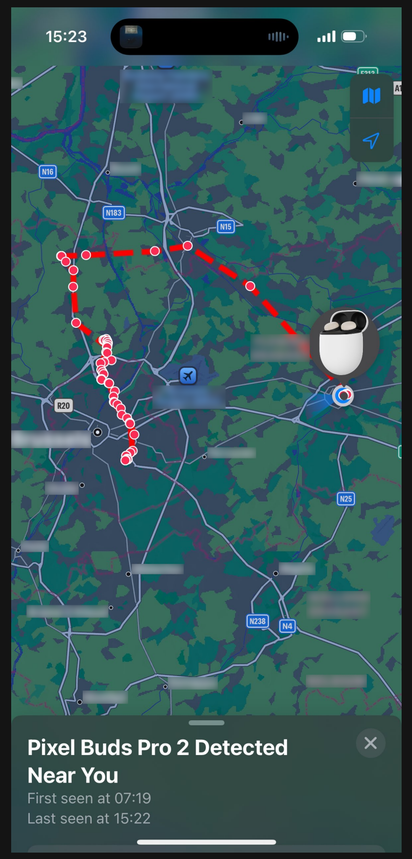
  → Enables long term tracking

Covert Location Tracking

Find Hub



Attacker: Find Hub location report

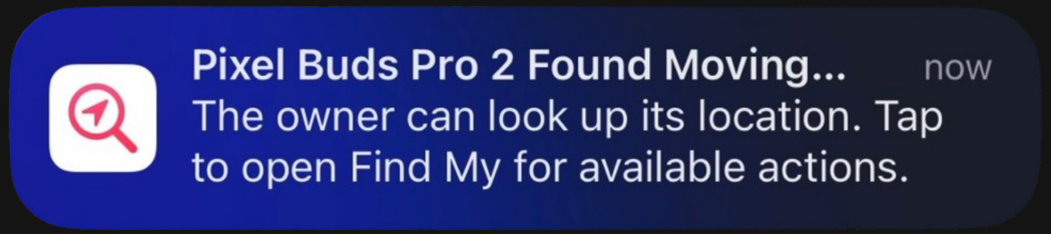


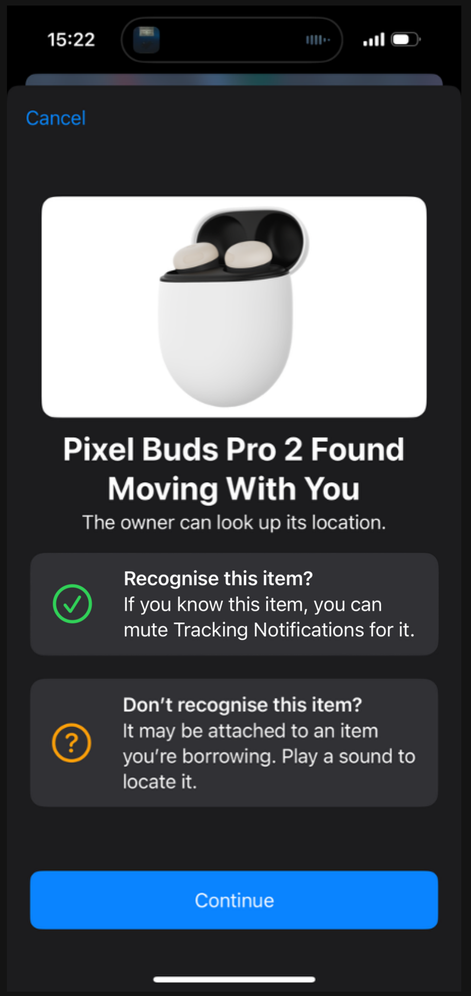
Victim: Anti-stalking notification

Unwanted Tracking Protection

Find Hub

- Apple and Android devices have built-in protections against stalking

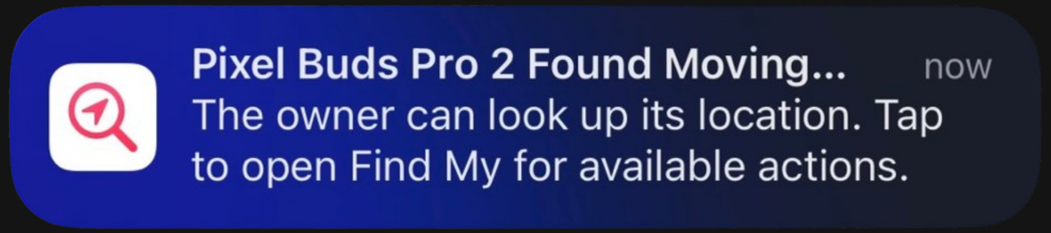


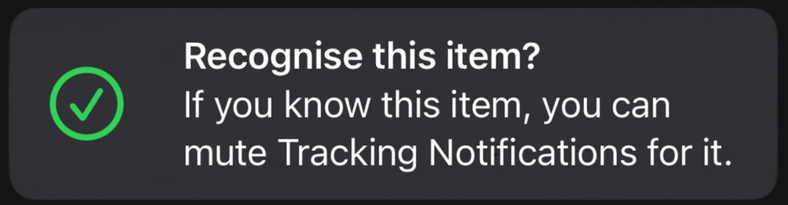


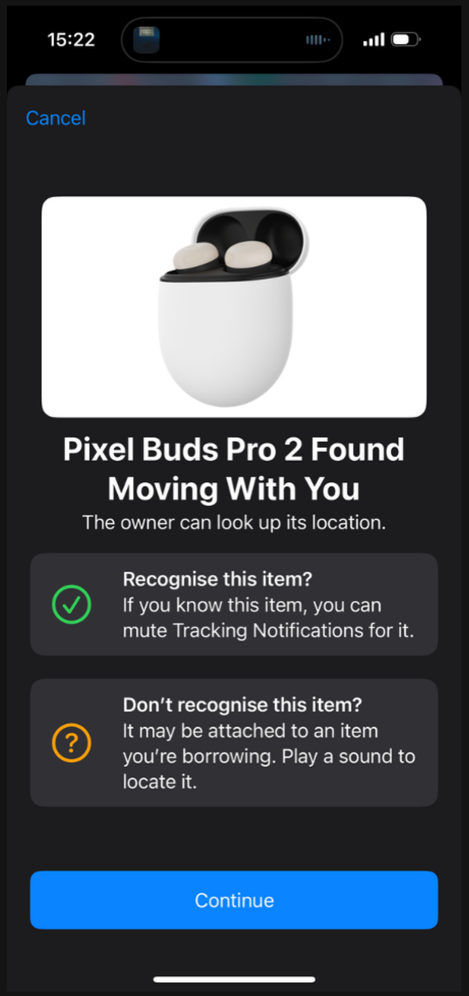
Unwanted Tracking Protection

Find Hub

- Apple and Android devices have built-in protections against stalking





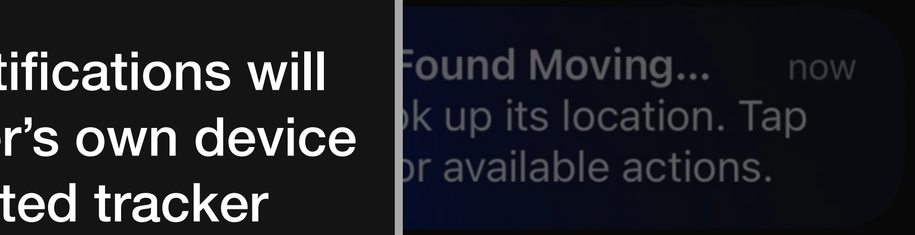


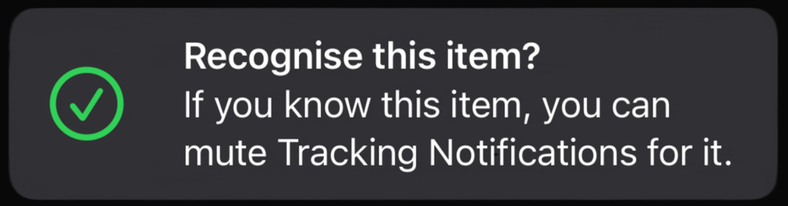
Unwanted Tracking Protection

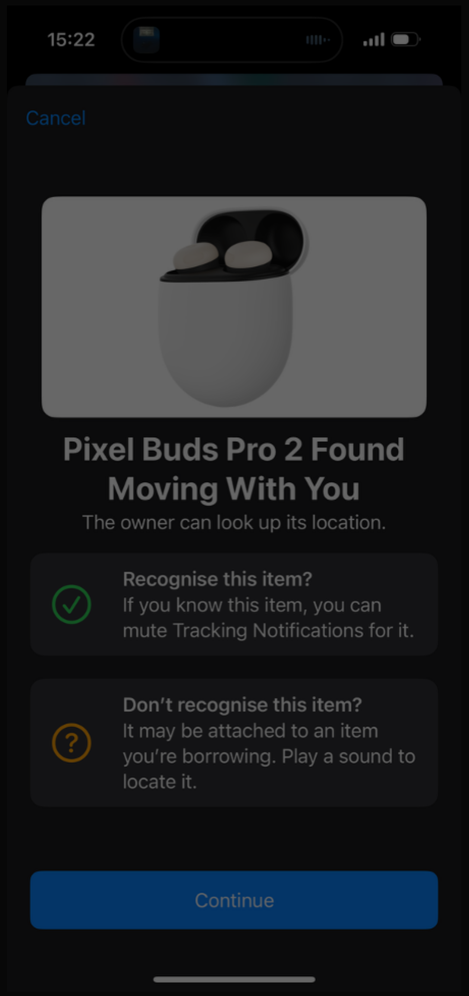
Find Hub

- Apple and Android devices have built-in protections against stalking

Tracking notifications will show the user's own device as unwanted tracker





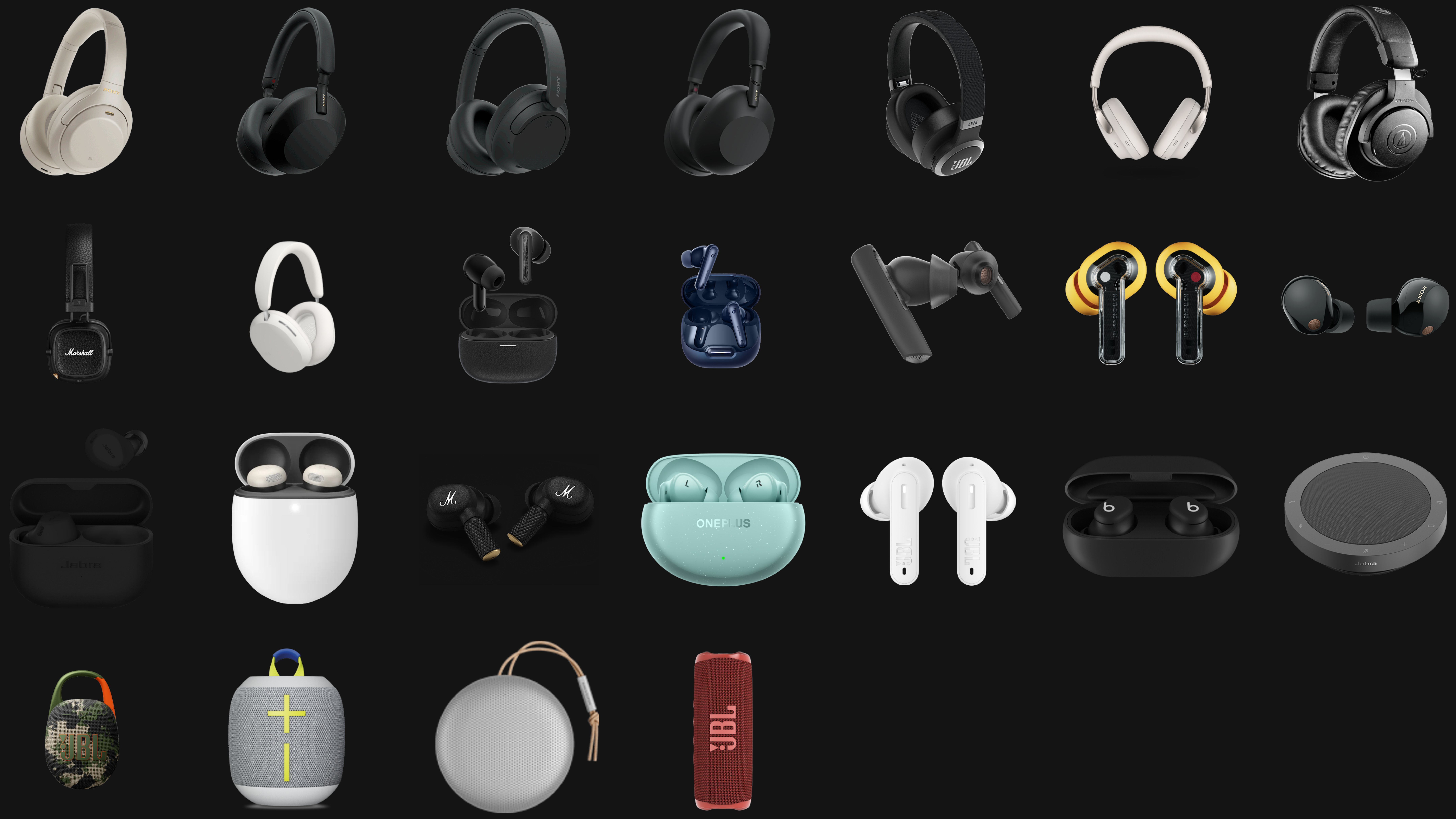


Evaluation

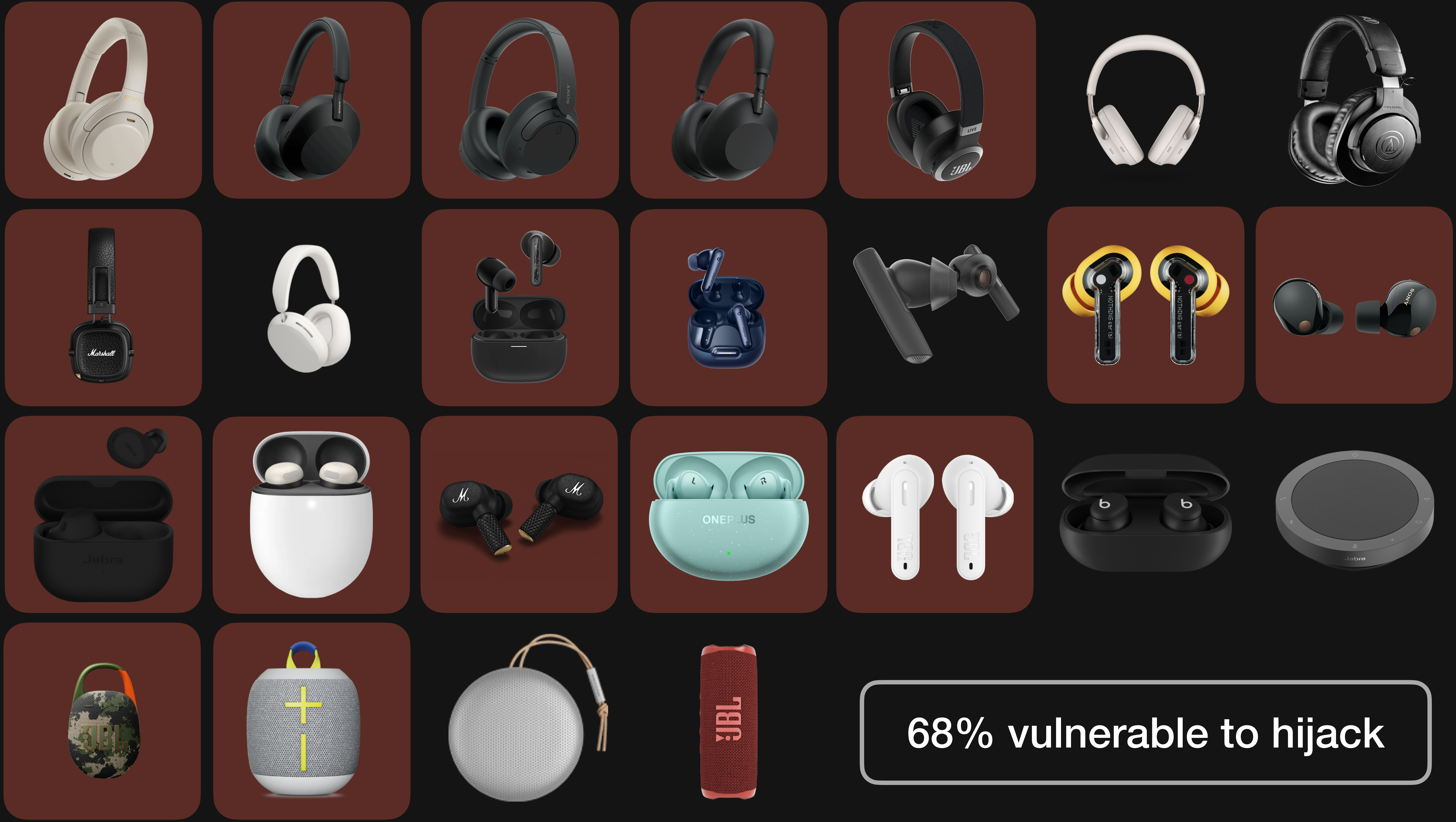
Experimental setup

- 25 devices from 16 different vendors
- Short distance (< 1 m) and long distance (14 m)

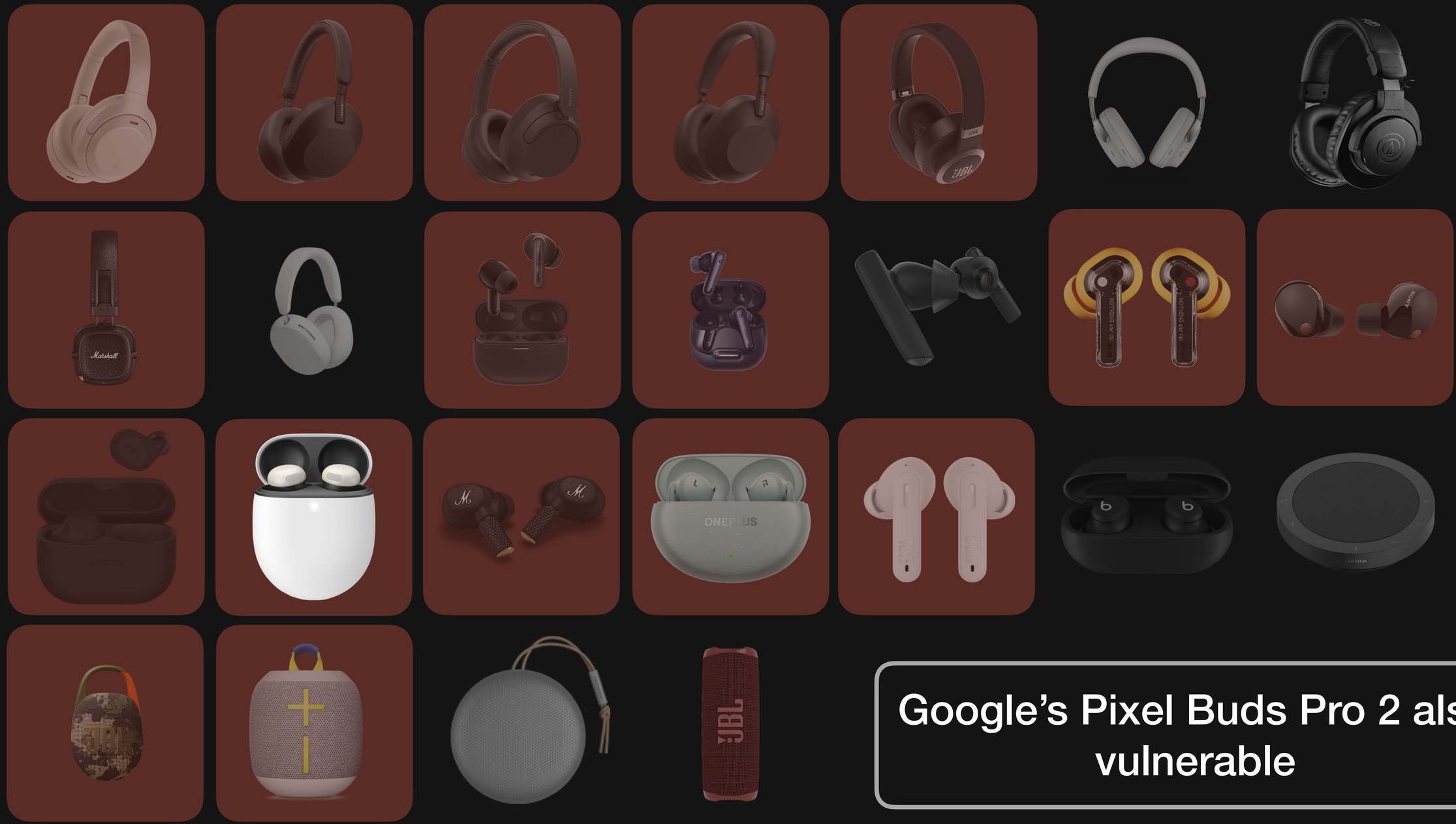
- **Measurements:**
  - Specification compliance
  - Time-to-hijack
  - Capabilities (Find Hub, Audio Switch)



JBL
Jabra
Marshall



68% vulnerable to hijack



Google’s Pixel Buds Pro 2 also vulnerable

Root Cause Analysis

Google Fast Pair

- The same class of failures appears across multiple vendors and chipset families
- This points to a **systemic issue** rather than a single implementation bug
- We investigated the root causes

Compliance Chain Failures

Google Fast Pair

- Vendors adopt Fast Pair through a multi step compliance workflow defined by Google
- The workflow has three stages:

1. **Implementation** by chipset and accessory vendors
2. **Validation** using Google’s Validator App and pass reports
3. **Certification** through Google approved conformance testing

- We identified **security critical failures** in all three stages

IntentPair

Can we eliminate the flaw by design?

IntentPair

Hardening mechanism

- **Small** protocol modification
- **Cryptographically binds** the pairing intent 🔒
- Advertise a nonce and use it to derive the session key

IntentPair

Hardening mechanism

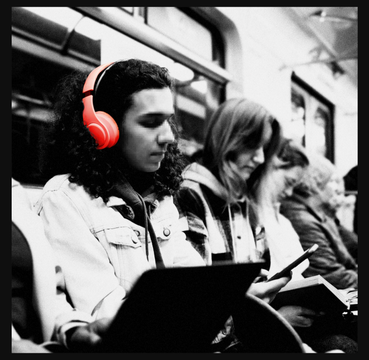
- **Small** protocol modification
- **Cryptographically binds** the pairing intent 🔒
- Advertise a nonce and use it to derive the session key

- Goal: *fail closed* by construction
  - Reduces reliance on **vendor specific** enforcement

Remediation

Disclosure process

- Reported to Google in August 2025
- Notifying partners, **releasing patches** would take time
- Disclosure timeline **extended to 150 days**
- Vulnerability rated **critical**, $15,000 bounty



ANDY GREENBERG | LILY HAY NEWMAN | SECURITY

JAN 15, 2026 7:00 AM

Hundreds of Millions of Audio Devices Need a Patch to Prevent Wireless Hacking and Tracking

Flaws in how 17 models of headphones and speakers use Google's one-tap Fast Pair Bluetooth protocol have left devices open to eavesdroppers and stalkers.

TECH GADGETS NEWS

Sony, Anker, and other headphones have a serious Google Fast Pair security vulnerability

/ Researchers say the issue can allow attackers to listen to the mics on wireless audio devices, or track their location.

by Andrew Liszewski

Jan 16, 2026, 3:13 PM GMT+1

Sony's WH-1000XM6 are among the wireless headphones affected by the Fast Pair vulnerability. Photo by Amelia Holowaty Krales / The Verge

Comments (All New)

ELECTRONICS › AUDIO

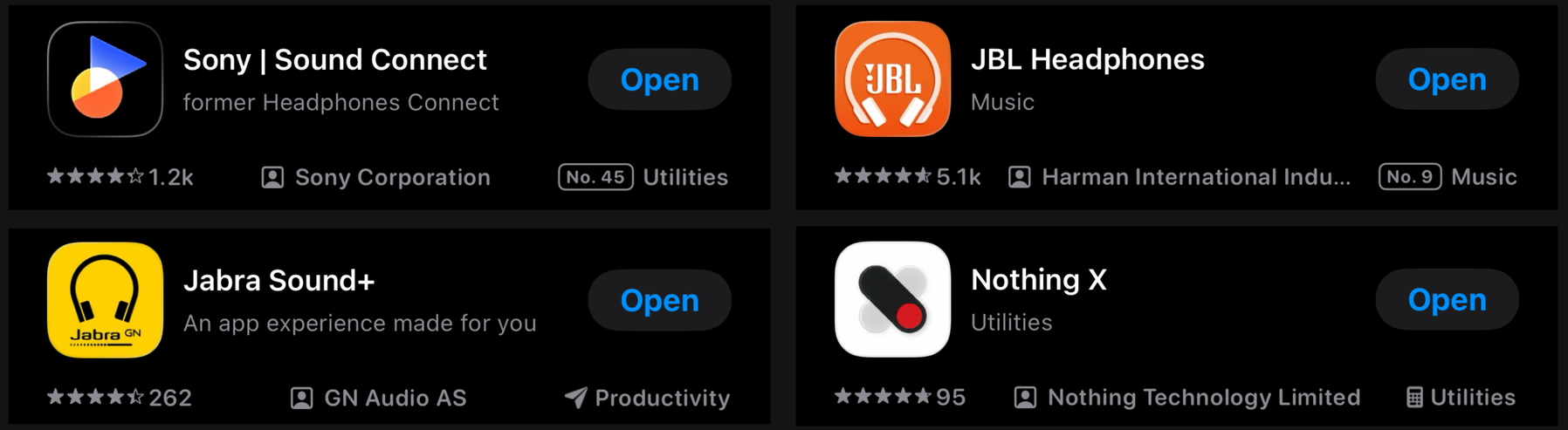
Wireless Earbuds Can Be Hacked. Here's How to Protect Yourself.

Published January 16, 2026

Remediation

Firmware updates

- Remediation requires a **firmware update** of the accessory
- **Cumbersome** process
  - Requires the manufacturer’s app



pi-fp.local

WhisperPair UI

Re-sort

Fast Pair devices

Pixel Buds Pro 2
75:A1:BE:C7:39:71
-39 Select
Fast pair | Model ID available | Connected

Unknown
44:52:01:B7:03:D9
-56 Select
Fast pair | Advertising | Aireware LLC

Unknown
C8:1C:36:9D:A3:74
-75 Select
Fast pair | Advertising

Paired devices

Other devices

LE_WH-1000XM4
C2:C3:40:15:25:93
-1 Select

Unknown
D2:99:E2:B1:60:27
-1 Select

Unknown
DD:DC:B0:C5:A7:72
-46 Select
Apple, Inc.

Pixel Buds Pro 2

75:A1:BE:C7:39:71

No active tasks

Disconnect | Attack | Set Model ID

Connected
`true`

Paired
`false`

Manufacturer
No data available

Fast Pair Data
Account Key List: `000033e4e45a`

Manual

Model ID
(...26...5c79)
Name
Pixel Buds Pro 2
Company name
Google
Device type
TRUE_WIRELESS_HEADPHONES
Features
6, EDDYSTONE_TRACKING, 12
Internal name
Porcelain

Demo

Fast Pair GATT Characteristics

Read Model ID

Model ID
fe2c1233-8366-4814-8eb0-01de32100bea
Key-based pairing
fe2c1234-8366-4814-8eb0-01de32100bea

One Tap To Hijack Them All

Recap

- **Systemic** failures in enforcing Fast Pair's core security requirements
  - **Incorrectly implemented** by manufacturers ⊗
  - Not detected during **certification** ☑

One Tap To Hijack Them All

Recap

- **Systemic** failures in enforcing Fast Pair's core security requirements
  - **Incorrectly implemented** by manufacturers
  - Not detected during **certification**

- Practical attacks:
  - **Hijacking** Bluetooth connections
  - **Stalking** using the Find Hub network

Conclusion
Key takeaways

- Even a small *add-on* can introduce **major privacy and security risks**
- If a problem can be solved at the **top**, it should be solved at the **top**
- **Update** your wireless headphones or earbuds

- More information: whisperpair.eu
- Will be presented at IEEE S&P 2026


black hat®
ASIA 2026
APRIL 21-24, 2026
MARINA BAY SANDS / SINGAPORE