



# Bluetooth Headphone Jacking: A Key to Your Phone

## Hacking Airoha-based Bluetooth Audio Devices



## About us



**Dennis Heinze**  
Security Researcher &  
Penetration Tester



**Frieder Steinmetz**  
Security Researcher &  
Penetration Tester

**Team Mobile and IoT Security @ ERNW**

Previously:



**Bluetooth Auracast Research.  
Goal: Obtain firmware and analyze implementation.**



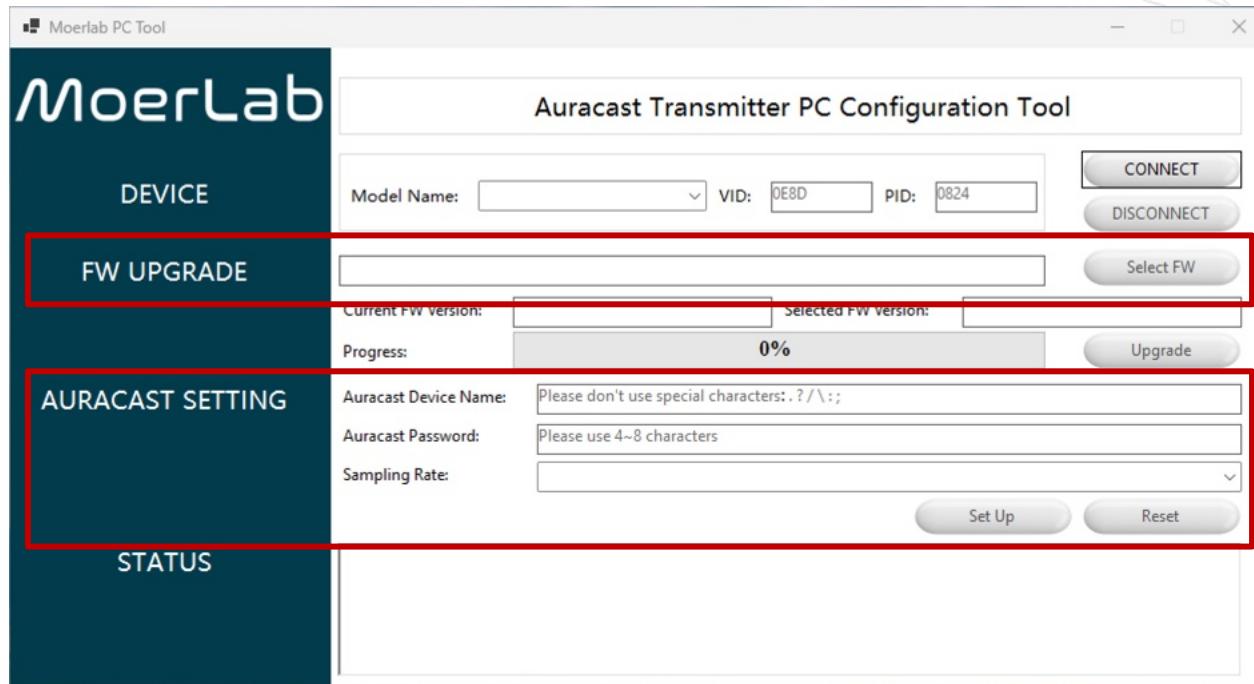
We have a firmware update file for this thing.

```
00000000: bb3b a797 eccc 2084 3272 2c7b c39f bbe8 .;.... .2r,{....  
00000010: 6af3 ba00 134d 7a76 3b25 21fe 4012 c002 j....Mzv;%!.@....  
00000020: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000030: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000040: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000050: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000060: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000070: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000080: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000090: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000a0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000b0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000c0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000d0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000e0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000f0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000100: 1100 0a00 0201 0010 0000 7016 1300 1300 .....p....  
00000110: 1c00 5635 2e32 2e32 00ff ffff ffff ffff ..V5.2.2.....  
00000120: ffff ffff ffff ffff ffff ffff ffff 1200 .....0  
00000130: 1c00 0200 0000 0010 0000 00f0 0800 0030 .....@....  
00000140: 0100 0000 0900 0040 1500 0020 0e00 1400 .....IM.lqm....  
00000150: 4400 0200 0000 494d ad6c 716d f400 ad12 D....m}!t....m8.7...  
00000160: 9a6d 7d21 74d8 ecf8 f36d 3889 37a4 b58f .m)!t....m8.7...
```

But it's a blob we don't know how to unpack.

Firmware  
Update →

Settings →



**USB-based tool for configuration and firmware updates.**

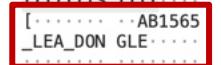
Apply a display filter ... <%> + 1 2 3

No.	Time	Source	Destination	Protocol	Length	Info
23	5.2308000	host	2.1.0	USB	72	GET_DESCRIPTOR Response STRING
24	5.231150	2.1.0	host	USB	36	GET_DESCRIPTOR Request STRING
25	5.367904	host	2.1.0	USB	62	GET_DESCRIPTOR Response STRING
26	5.369192	2.1.0	host	USB	36	GET_REPORT Request
27	5.370470	host	2.1.0	USB	90	GET_REPORT Response
28	5.372257	2.1.0	host	USB	98	SET_REPORT Request
29	5.375542	host	2.1.0	USB	28	SET_REPORT Response
30	5.377944	2.1.0	host	USB	36	GET_REPORT Request
→ 31	5.379046	host	2.1.0	USB	90	GET_REPORT Response
← 32	5.380170	2.1.0	host	USB	98	SET_REPORT Request
33	5.386652	host	2.1.0	USB	28	SET_REPORT Response
34	5.388613	2.1.0	host	USB	36	GET_REPORT Request
35	5.390878	host	2.1.0	USB	90	GET_REPORT Response
36	5.391723	2.1.0	host	USB	98	SET_REPORT Request
37	5.445078	host	2.1.0	USB	28	SET_REPORT Response
38	5.446923	2.1.0	host	USB	36	GET_REPORT Request
39	5.447941	host	2.1.0	USB	90	GET_REPORT Response
40	5.448903	2.1.0	host	USB	36	GET_REPORT Request
41	5.449036	host	2.1.0	USB	100	GET REPORT Request

> Frame 32: Packet, 90 bytes on wire (720 bits), 90 bytes captured (720 bits)  
 ↓ USB URB  
 [Source: 2.1.0]  
 [Destination: host]  
**USBPcap pseudoheader length: 28**  
 IRP ID: 0xffffe68b4478aa20  
 IRP USBD\_STATUS: USBD\_STATUS\_SUCCESS (0x00000000)  
 URB Function: URB\_FUNCTION\_CONTROL\_TRANSFER (0x0008)  
 > IRP information: 0x01, Direction: PDO → FDO  
 URB bus id: 2  
 Device address: 1  
 > Endpoint: 0x80, Direction: IN  
 URB transfer type: URB\_CONTROL (0x02)  
 Packet Data Length: 62  
 [Request in: 31]  
 [Time from request: 1.124000 milliseconds]  
 Control transfer stage: Complete (3)  
 [hInterfaceClass: HID (0x03)]

0000 1c 00 20 aa 78 44 8b e6 ff ff 00 00 00 00 08 00 ... xD .. ....  
 0010 01 02 00 01 00 80 02 3e 00 00 00 03 07 1c 00 05  
 0020 5b 18 00 0c 0a 00 02 10 11 00 41 42 31 35 36 35  
 0030 5f 4c 45 41 5f 44 4f 4e 47 4c 45 00 00 00 00 00 00  
 0040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
 0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

ASCII



No.	Time	Source	Destination	Protocol	Length	Info
→ 43391	172.220219	host	2.1.0	USBHID	98	SET_REPORT Request
← 43392	172.221029	2.1.0	host	USBHID	28	SET_REPORT Response
43393	172.222864	host	2.1.0	USBHID	98	SET_REPORT Request
43394	172.223951	2.1.0	host	USBHID	28	SET_REPORT Response
43395	172.225844	host	2.1.0	USBHID	98	SET_REPORT Request
43396	172.226695	2.1.0	host	USBHID	28	SET_REPORT Response
43397	172.227184	host	2.1.0	USBHID	98	SET_REPORT Request
43398	172.227884	2.1.0	host	USBHID	28	SET_REPORT Response
43399	172.229904	host	2.1.0	USBHID	98	SET_REPORT Request
43400	172.230547	2.1.0	host	USBHID	28	SET_REPORT Response
43401	172.230843	host	2.1.0	USBHID	98	SET_REPORT Request
43402	172.231620	2.1.0	host	USBHID	28	SET_REPORT Response
43403	172.231927	host	2.1.0	USBHID	98	SET_REPORT Request
43404	172.232742	2.1.0	host	USBHID	28	SET_REPORT Response
43405	172.234849	host	2.1.0	USBHID	98	SET_REPORT Request
43406	172.235404	2.1.0	host	USBHID	28	SET_REPORT Response
43407	172.235892	host	2.1.0	USBHID	98	SET_REPORT Request
43408	172.237269	2.1.0	host	USBHID	28	SET_REPORT Response

IRP USBD\_STATUS: USBD\_STATUS\_SUCCESS (0x00000000)  
URB Function: URB\_FUNCTION\_CLASS\_INTERFACE (0x001b)

- > IRP information: 0x00, Direction: FDO → PDO
- URB bus id: 2
- Device address: 1
- > Endpoint: 0x00, Direction: OUT
- URB transfer type: URB\_CONTROL (0x02)
- Packet Data Length: 70
- [Response in: 43392]
- Control transfer stage: Setup (0)
- [bInterfaceClass: HID (0x03)]

Setup Data

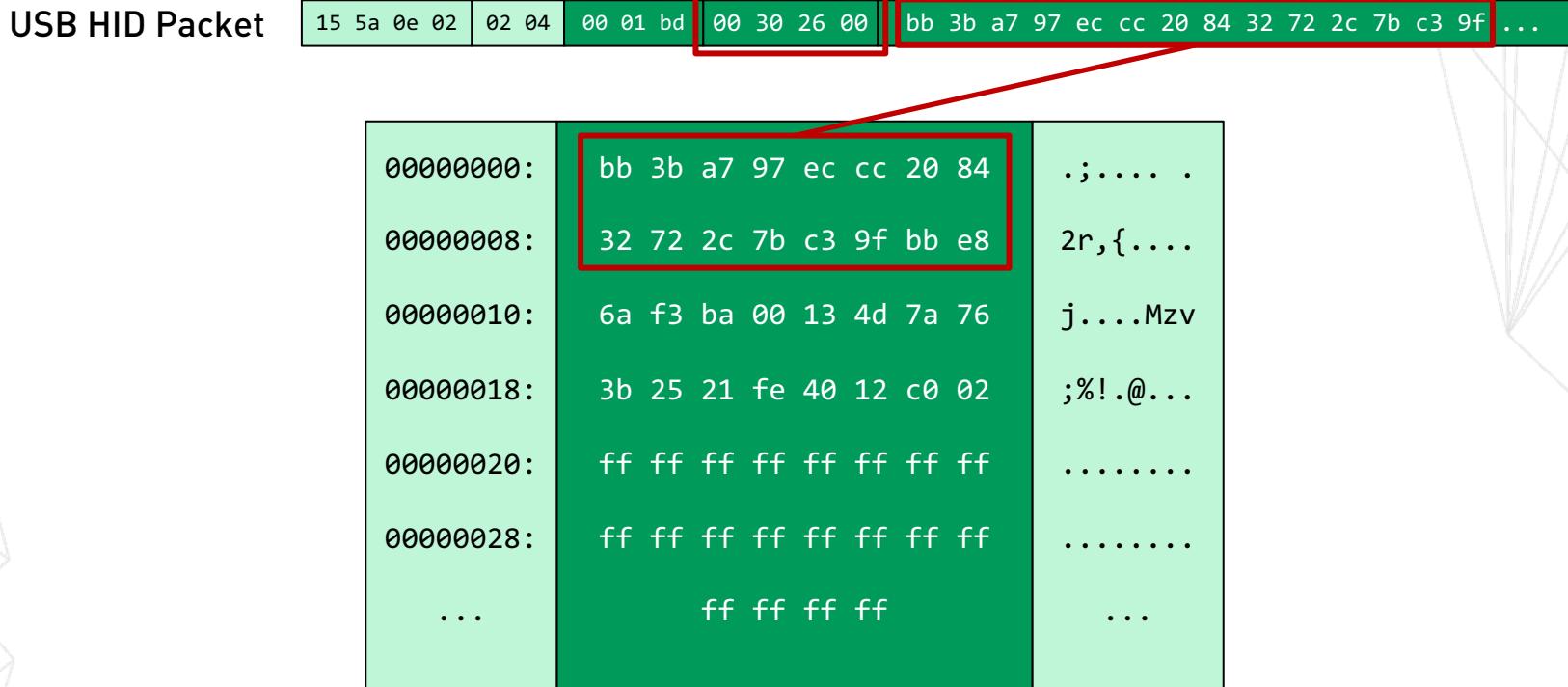
- > bmRequestType: 0x21
- bRequest: SET\_REPORT (0x09)
- > wValue: 0x0206
- wIndex: 3
- wLength: 62

Data Fragment: 063b00155a0e0202040002bd00302600bb3ba797ecc208432722c7bc

Packet (98 bytes) USB Control (69 bytes)

Binary blob

# Firmware Update Command?



# Firmware Update Command?

USB HID Packet



00000100:	11 00 0a 00 02 01 00 10	.....
00000108:	00 00 70 16 13 00 13 00	..p.....
00000110:	1c 00 56 35 2e 32 2e 32	..v5.2.2
00000118:	00 ff ff ff ff ff ff ff	.....
00000120:	ff ff ff ff ff ff ff ff	.....
00000128:	ff ff ff ff ff ff 12 00	.5.....I
...	1c 00 02 00	....

Firmware Update File

# Writing Data

15 5a 0e 02	<b>02 04</b>	00 01 bd	<b>00 30 26 00</b>	bb 3b a7 97 ec cc 20 84 32 72 2c 7b c3 9f ...
-------------	--------------	----------	--------------------	---

15 5a 0e 02	<b>02 04</b>	00 01 b7	<b>00 31 26 00</b>	11 00 0a 00 02 01 00 10 00 00 70 16 13 00 ...
-------------	--------------	----------	--------------------	---

• • •

15 5a 0e 02	<b>02 04</b>	00 01 b8	<b>00 40 26 00</b>	0e 4b a6 fa ff 83 67 67 f7 31 d4 b8 0f 3c ...
-------------	--------------	----------	--------------------	---

# In the meantime,...

```
00000000: bb3b a797 eccc 2084 3272 2c7b c39f bbe8 .;.... .2r,{....  
00000010: 6af3 ba00 134d 7a76 3b25 21fe 4012 c002 j....Mzv;%!@....  
00000020: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000030: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000040: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000050: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000060: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000070: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000080: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000090: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000a0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000b0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000c0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000d0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000e0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
000000f0: ffff ffff ffff ffff ffff ffff ffff ffff .....  
00000100: 1100 0a00 0201 0010 0000 7016 1300 1300 .....p....  
00000110: 1c00 5635 2e32 2e32 00ff ffff ffff ffff ..V5.2.2.....  
00000120: ffff ffff ffff ffff ffff ffff ffff 1200 .....  
00000130: 1c00 0200 0000 0010 0000 00f0 0800 0030 .....0....  
00000140: 0100 0000 0900 0040 1500 0020 0e00 1400 .....@....  
00000150: 4400 0200 0000 494d ad6c 716d f400 ad12 D.....IM.1qm....  
00000160: 9a6d 7d21 74d8 ecf4 f36d 3889 37a4 b58f .m}!t....m8.7...
```

Now we know how to unpack the blob!



LZMA  
(and potentially decryption)



<https://github.com/ramikg/airoha-firmware-parser>

# Following the Command ID: Write Flash

```
switch(cmd_id) {  
    case 0x400:  
        pvVar3 = FUN_0838bad4(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
    case 0x402:  
        pvVar3 = cmd_write_flash(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
    case 0x403:  
        pvVar3 = FUN_0838b920(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
    case 0x404:  
        pvVar3 = FUN_0838b994(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
}
```

# Following the Command ID: Read Flash

```
switch(cmd_id) {  
    case 0x400:  
        pvVar3 = FUN_0838bad4(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
    case 0x402:  
        pvVar3 = cmd_write_flash(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
    case 0x403:  
        pvVar3 = cmd_read_flash(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
    case 0x404:  
        pvVar3 = FUN_0838b994(&inputMsg->hdr,length,transport_id);  
        return pvVar3;  
}
```

# Found the Flash Read Command

- We built a tool and implemented the protocol.
- We were able to dump flash via USB HID.
- And it worked for the headphones too!

At this point we were hooked and kind of forgot about our initial goal, the Auracast research.

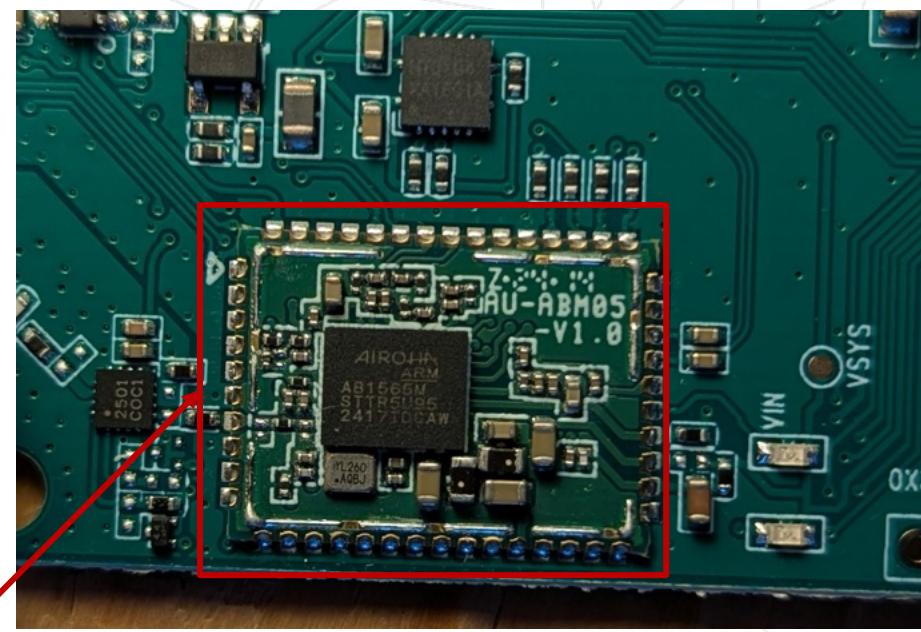
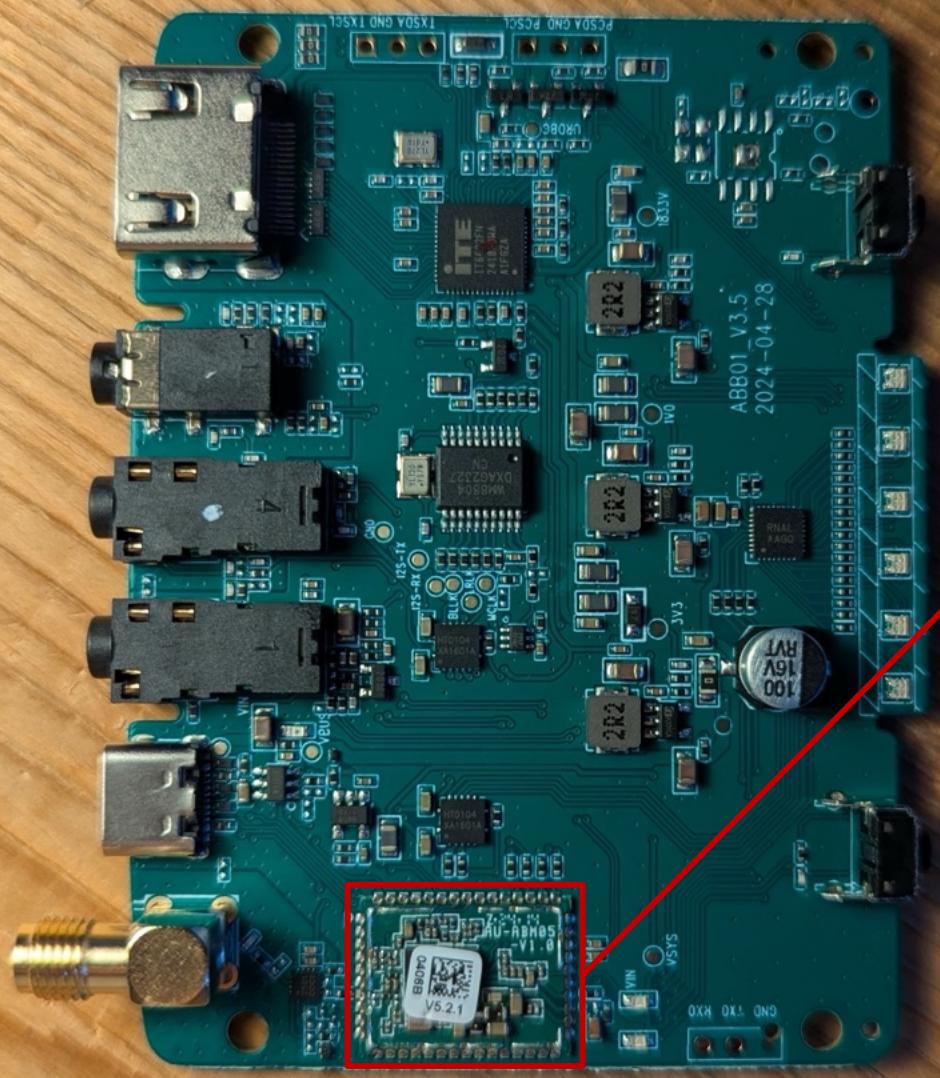


# Does that work with other, more prestigious devices?



Like these \$9.000 Bose  
diamond crusted  
earbuds?





What do the headphones and transmitter have in common?

**AIROHA**

達發科技股份有限公司

AB1565M

## Search the internet

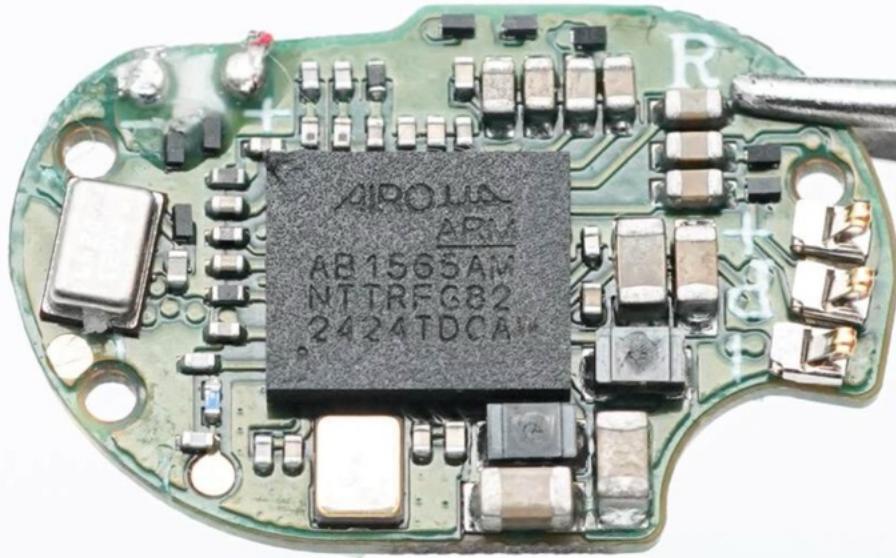
A screenshot of a web browser window. The address bar shows a secure connection to <https://english.cw.com.tw/article/article.action?id=3257>. The page header includes a menu icon, the logo for "天下雜誌 Commonwealth Magazine" (with the Chinese characters 天下 in red), a search input field with the placeholder "keyword", and a magnifying glass icon. The main content area displays a news article snippet: "In 2019, Airoha's [REDACTED] TWS chips that were used in Sony's newest noise-canceling true wireless earbuds, the WF-1000XM3."

In 2019, Airoha's [REDACTED] TWS chips that were used in Sony's newest noise-canceling true wireless earbuds, the WF-1000XM3.

连接通话麦克风小板的金属弹片特写。

52audio.com

For extremely detailed  
teardowns and awesome  
pictures!



52AUDIO

AIROHA达发（络达）AB1565AM蓝牙音频SoC，AB1565x系列是Airoha蓝牙进阶款音讯解决方案，支持蓝牙5.3，支持LE audio，新的“Dual Link TWO”，可同时连接笔记本和手机轻松切换，提供超低功耗和超低延时表现。平均功耗可达4~4.5mA，为目前



r/AirReps

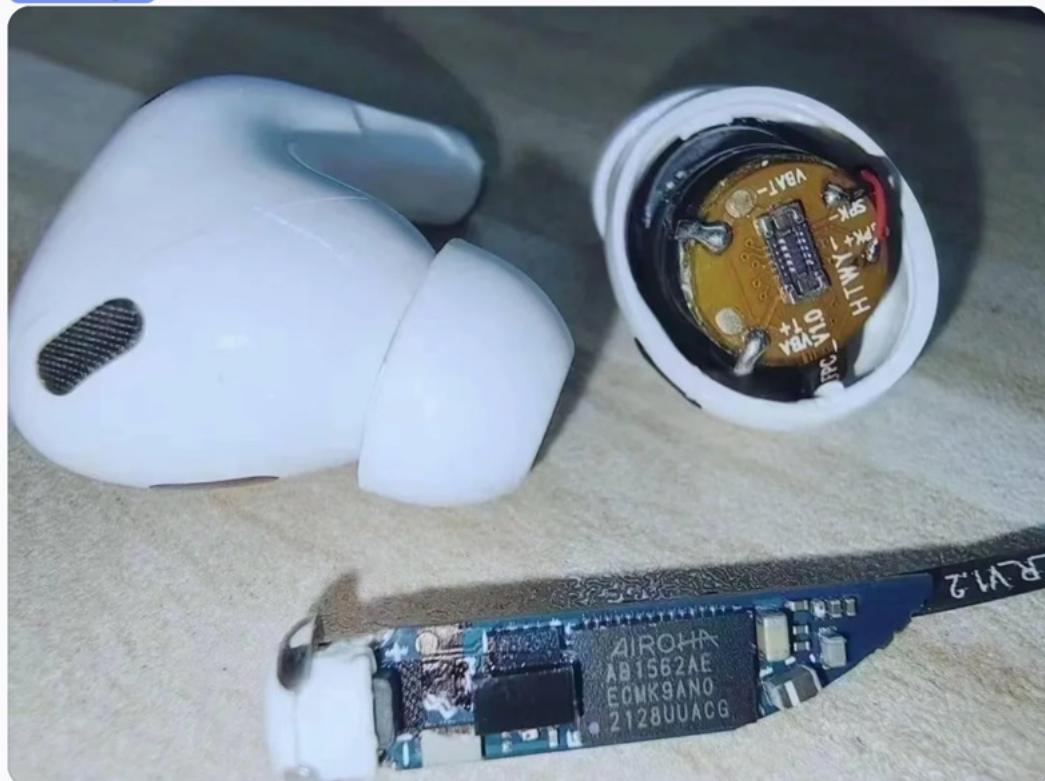


Suche in r/AirReps

r/AirReps · vor 1 Jahr  
FabSa01

## Is it real Airoha 1562AE?

Discussion



## AirPods replicas

There is an active community around Airoha-based AirPods replicas.

In fact, there are plenty of fancy devices  
with Airoha Bluetooth chips.

**For example:**

Sony  
Marshall  
Jabra

JBL  
Beyerdynamic  
Bose

**So, we bought some.**

Some of our test device arsenal. Stay tuned for our headphone review  
YouTube channel!



# No USB

## But how do you update the headphones?



There's an app for that!

File | New | Open | Save | Cut | Copy | Paste | Find | Replace | Select All | Exit

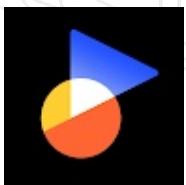
Project: airoha

- airoha
  - libbase
    - constant
      - AgentPartnerEnum
      - StopSenderEnum
    - interfaceMgr
      - AirohaMgrInterface
    - RaceCommand
      - constant
        - AppId
          - FOTA byte
          - MMI byte
          - AppId() void
      - AvailableDst
        - RACE\_CHANNEL\_TYPE\_AIRAPP byte
        - RACE\_CHANNEL\_TYPE\_AIRUPDATE byte
        - RACE\_CHANNEL\_TYPE\_ANSPEER byte
        - RACE\_CHANNEL\_TYPE\_SOFTWARE byte
        - RACE\_CHANNEL\_TYPE\_UART byte
        - RACE\_CHANNEL\_TYPE\_USB byte
      - AvailableDst() void
    - FotaModeId
      - SVALUES FotaModeId[]
        - Active FotaModeId
        - Adaptive FotaModeId
        - Background FotaModeId
        - mName String
        - mValue int
        - {...} void
      - FotaModeId(String, int, int, String) void
      - getName() String
      - getValue() int
      - valueOf(String) FotaModeId
      - values() FotaModeId[]
    - NvKeyId
    - RaceId
    - RaceType
    - Recipient
    - packet
    - relay
    - BuildConfig
    - R
  - libcommon
  - libfota155x
  - libfota1562
  - libfota1568
  - liblinker
  - liblogdump
  - liblogger
  - libmmi
  - libmmi1562
  - libmmi1568
  - libNativeAnc
  - libNativeAnc1568
  - libNativePeq
  - libpeq
  - libutils
  - sdk
  - utapp.sdk
  - WIFI

```

1 package comairoha.libbase.RaceCommand.constant;
2
3 /* loaded from: classes.dex */
4 public class RaceId {
5     public static final int DSP_REALTIME_ANC_ADAPTIVE_CHECK = 3603;
6     public static final int DSP_REALTIME_ANC_ADAPTIVE_RESULT = 3604;
7     public static final int DSP_REALTIME_ANC_ON = 3590;
8     public static final int DSP_REALTIME_ANC_SET_RUNTIME_VOL = 3590;
9     public static final int RACE_AIRDUMP_ONOFF = 3587;
10    public static final int RACE_ANC_DNDFF = 3595;
11    public static final int RACE_ANC_GET_STATUS = 4616;
12    public static final int RACE_ANC_OFF = 4609;
13    public static final int RACE_ANC_ON = 4608;
14    public static final int RACE_ANC_READ_PARAM_FROM_NVKEY = 4612;
15    public static final int RACE_ANC_SET_GAIN = 4611;
16    public static final int RACE_ANC_WRITE_GAIN_TO_NVKEY = 4613;
17    public static final int RACE_ANTENNAUT_REPORT_ENABLE = 5888;
18    public static final int RACE_BLUETOOTH_GET_BOX_BATTERY = 31;
19    public static final int RACE_BLUETOOTH_GET_CHARGING_INFO = 9;
20    public static final int RACE_BLUETOOTH_GET_CLIENT_EXISTENCE = 3283;
21    public static final int RACE_BLUETOOTH_GET_IR_STATUS = 12;
22    public static final int RACE_BLUETOOTH_IS_AGENT_RIGHT_DEVICE = 3284;
23    public static final int RACE_BLUETOOTH_ROLE_SWITCH = 3287;
24    public static final int RACE_BLUETOOTH_TWS_GET_BATTERY = 3286;
25    public static final int RACE_FIND_ME = 11265;
26    public static final int RACE_FIND_ME_QUERY_STATE = 11264;
27    public static final int RACE_FLASH_DUAL_DEVICES_PARTITION_ERASE = 1807;
28    public static final int RACE_FLASH_PAGE_PROGRAM = 1799;
29    public static final int RACE_FLASH_SECTOR_ERASE = 1796;
30    public static final int RACE_FLASH_SET_PROTECT_BIT = 1794;
31    public static final int RACE_FOTA_ACTIVE_FOTA_PREPARATION = 7193;
32    public static final int RACE_FOTA_COMMIT = 7178;
33    public static final int RACE_FOTA_DUAL_DEVICES_CANCEL = 7189;
34    public static final int RACE_FOTA_DUAL_DEVICES_COMMIT = 7185;
35    public static final int RACE_FOTA_DUAL_DEVICES_QUERY_PARTITION_INFO = 7188;
36    public static final int RACE_FOTA_DUAL_DEVICES_QUERY_STATE = 7186;
37    public static final int RACE_FOTA_DUAL_DEVICES_START_TRANSACTION = 7184;
38    public static final int RACE_FOTA_DUAL_DEVICES_WRITE_STATE = 7187;
39    public static final int RACE_FOTA_GET_A2DP_CONNECTION_INFO = 8;
40    public static final int RACE_FOTA_GET_AE_INFO = 7177;
41    public static final int RACE_FOTA_GET_INTERNAL_FLASH_PARTITION_SHA256 = 7183;
42    public static final int RACE_FOTA_GET_PARTITION_ERASE_STATUS = 7190;
43    public static final int RACE_FOTA_GET_ROFS_VERSION = 7426;
44    public static final int RACE_FOTA_GET_VERSION = 7175;
45    public static final int RACE_FOTA_INTEGRITY_CHECK = 7169;
46    public static final int RACE_FOTA_PARTITION_INFO_QUERY = 7168;
47    public static final int RACE_FOTA_PING = 7195;
48    public static final int RACE_FOTA_QUERY_STATE = 7172;
49    public static final int RACE_FOTA_SET_IR_ONOFF = 10;
50    public static final int RACE_FOTA_SET_POWER_CONTROL_INFO = 7;
51    public static final int RACE_FOTA_START = 7176;
52    public static final int RACE_FOTA_START_TRANSACTION = 7178;
53    public static final int RACE_FOTA_STOP = 7171;
54    public static final int RACE_FOTA_WRITE_STATE = 7174;
55    public static final int RACE_GET_A2DP_PARAMETER = 3279;
56    public static final int RACE_GET_ANC_SPORTS_MODE_STATUS = 13;
57    public static final int RACE_GET_BATTERY = 3074;
58    public static final int RACE_GET_BOOT_REASON = 7688;
59    public static final int RACE_GET_BUILD_VERSION_INFO = 7688;
60    public static final int RACE_GET_DEVICE_COLOR_INDEX = 14;
61    public static final int RACE_GET_DUMP_ADDR = 7682;
62    public static final int RACE_GET_DUMP_REGION_INFO = 7684;
63    public static final int RACE_GET_EXCEPTION_LOG_ADDR = 7686;
64    public static final int RACE_GET_IN_EAR_ONOFF = 11281;
65    public static final int RACE_GET_LE_LINK_STATUS = 3282;
66    public static final int RACE_GET_MHT_COMMON_CONFIG = 11395;
67    public static final int RACE_GET_OFFLINE_LOG_INFO = 7685;
68    public static final int RACE_GET_POWER_MODE = 524;
69    public static final int RACE_GET_SOFT_VERSION = 760;
70    public static final int RACE_HOSTAUDIO_MMI_GET_ENUM = 2305;
71    public static final int RACE_HOSTAUDIO_MMI_SET_ENUM = 2304;
72    public static final int RACE_HOSTAUDIO_PEQ_SAVE_STATUS = 2557;
73    public static final int RACE_HOSTAUDIO_PEQ_THRESHOLD = 2558;

```



Sony SoundConnect



AirReps 156X App

...



The protocol is called:

# RACE

## Quick Interlude: Bluetooth Classic vs. BLE

Bluetooth Classic	Bluetooth Low Energy
Built for high data rate (e.g. audio)	Built for energy efficiency
Public device address	Public/Random addresses for privacy
Discoverable vs. Non-discoverable	Advertisements
RFCOMM/L2CAP	GATT

# Bluetooth Classic as Transport

- RACE uses the RFCOMM profile as transport.
- RFCOMM is a Bluetooth Classic protocol, emulates serial interface
- RFCOMM has channels that are identified by their channel number
- Channel numbers can be discovered using the SDP protocol and a channel UUID.
- Channel UUIDs for RACE are vendor specific.

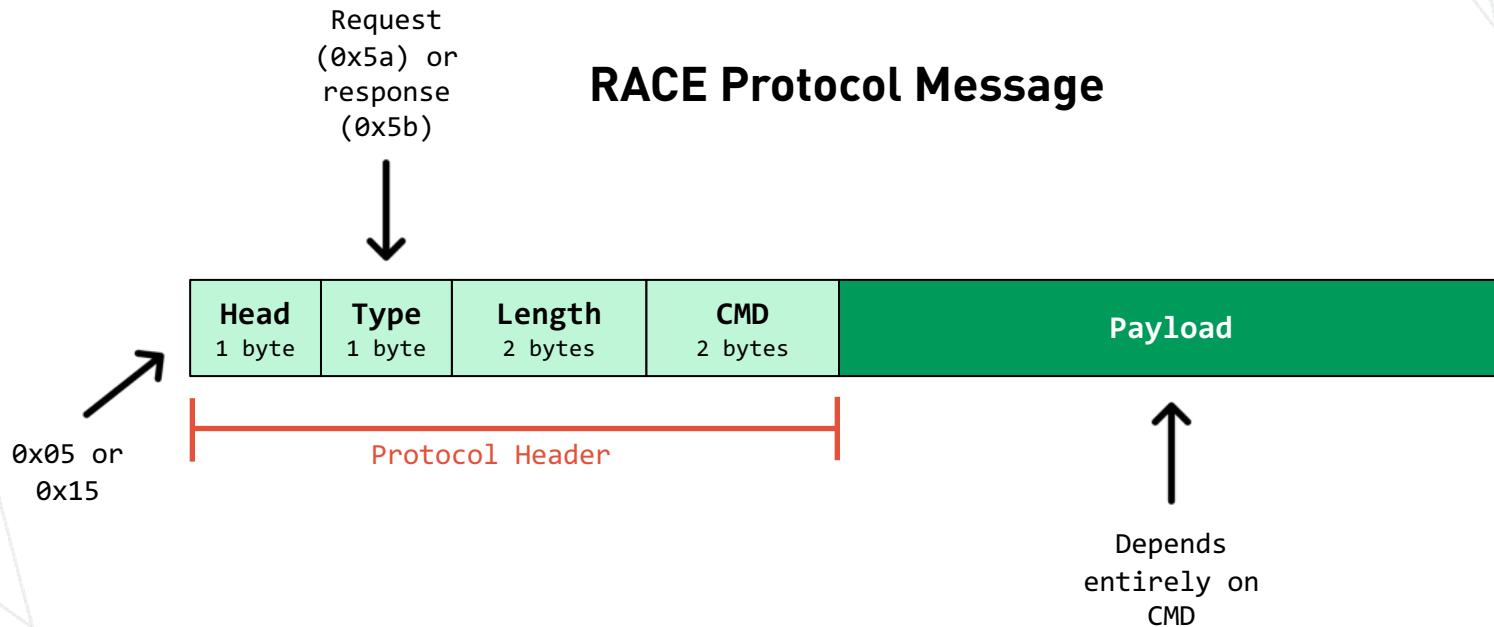
```
COMMON_SPP_UUID =      UUID("00001101-0000-1000-8000-00805F9B34FB")
AIROHA_SPP_UUID =     UUID("00000000-0000-0000-0099-AABBCCDDEEFF")
SONY_SPP_UUID =       UUID("8901DFA8-5C7E-4D8F-9F0C-C2B70683F5F0")
BOSE_SPP_UUID =        UUID("2D064AA9-32B5-4970-865C-643742BD2862")
BEYER_SPP_UUID =      UUID("7B46E8DE-5A7E-4512-BE0D-863AEAA3312B")
```

# Bluetooth Low Energy as Transport

- RACE is also available via GATT.
- GATT has services.
- Services have characteristics.
- Characteristics can be read from, written and subscribed to.
- Services & Characteristics can be discovered and have UUIDs.
- UUIDs for RACE are sometimes vendor specific.

```
AIROHA_GATT_SERVICE_UUID =     UUID("5052494D-2DAB-0341-6972-6F6861424C45")
AIROHA_GATT_TX_UUID =          UUID("43484152-2DAB-3241-6972-6F6861424C45")
AIROHA_GATT_RX_UUID =          UUID("43484152-2DAB-3141-6972-6F6861424C45")
SONY_GATT_SERVICE_UUID =        UUID("dc405470-a351-4a59-97d8-2e2e3b207fbb")
SONY_GATT_TX_UUID =             UUID("bfd869fa-a3f2-4c2f-bcff-3eb1ec80cead")
SONY_GATT_RX_UUID =             UUID("2a6b6575-faf6-418c-923f-cccd63a56d955")
```

## RACE Protocol Message



## RACE: Read Flash

<b>Head</b> 1 byte	<b>Type</b> 1 byte	<b>Length</b> 2 bytes	<b>CMD</b> 2 bytes	<b>Payload</b>		
0x05	0x5a	0x0008	0x0403	0x00	0x01	0x08000000
				<b>Type</b> 1 byte	<b>Num</b> 1 byte	<b>Address</b> 4 bytes

## RACE: Get Build Version

<b>Head</b> 1 byte	<b>Type</b> 1 byte	<b>Length</b> 2 bytes	<b>CMD</b> 2 bytes	<b>Payload</b>
0x05	0x5a	0x0008	0x1E08	-

## Things we know RACE can do:

- Read (and write\*) flash.
- Read and write RAM/register/peripheral (bytewise from any address).
- FOTA (Firmware Over the Air) operations.
- Get Bluetooth Classic device address.
- Read out build version.
- ... many other commands!
  
- Not every device supports all these functions.

# Tool Time

```
sudo python race_toolkit.py --transport rfcomm --target-address -c usb:0
$ sudo python race_toolkit.py --transport rfcomm --target-address 80:99:E7:37:49:50 -c usb:0 flash --address 0x00 --size 0x20000
Device initialized.
Found device with class SONY_WH - WH-1000XM5 - 80:99:E7:37:49:50/P
Channel found: 21
Connected to 80:99:E7:37:49:50 on channel 21
Dumping Flash: 38% | 197/512 [00:05<00:08, 37.36page/s]
```

- We built a tool that implements the RACE protocol via RFCOMM and GATT.
- Using Google's Python-based Bluetooth library Bumble.
- This is nice for research on our headphones: Dumping flash, debugging via RAM read/write.



The background features a minimalist abstract design composed of thin, light gray lines forming a complex network of triangles and polygons. These shapes are primarily located in the upper right and lower left quadrants, creating a sense of depth and perspective.

**But then we realized something...**

# Umm...

- During testing we always put our devices in pairing mode before connecting.
- Turns out: pairing is not required. No authentication.
- Anyone in Bluetooth range can connect to our headphones!
- Anyone in Bluetooth range can dump our headphones' flash.
- Anyone in Bluetooth range can read & write our headphones' RAM.



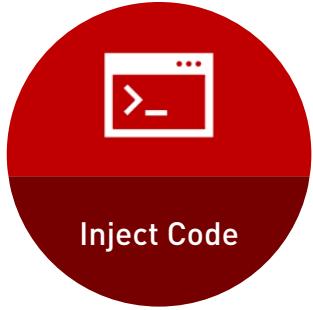
# Live Demos!



Extract data



Eavesdrop



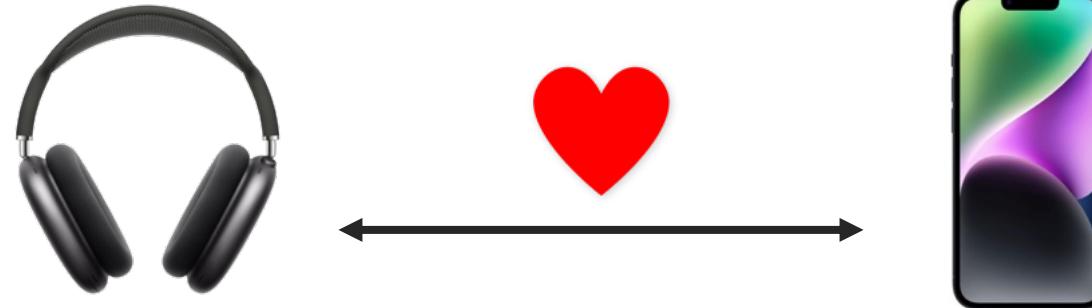
Inject Code



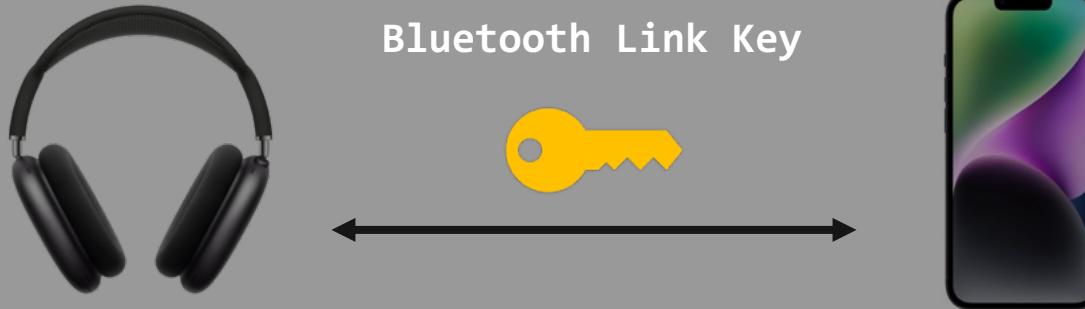
Pivoting



**At the center there's this  
love relationship**



**Or rather:  
This cryptographic bond**



## What do people use headphones for?



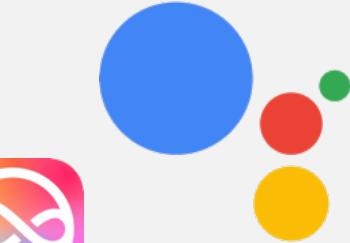
Spotify®



music and audio content



calls and voice messages



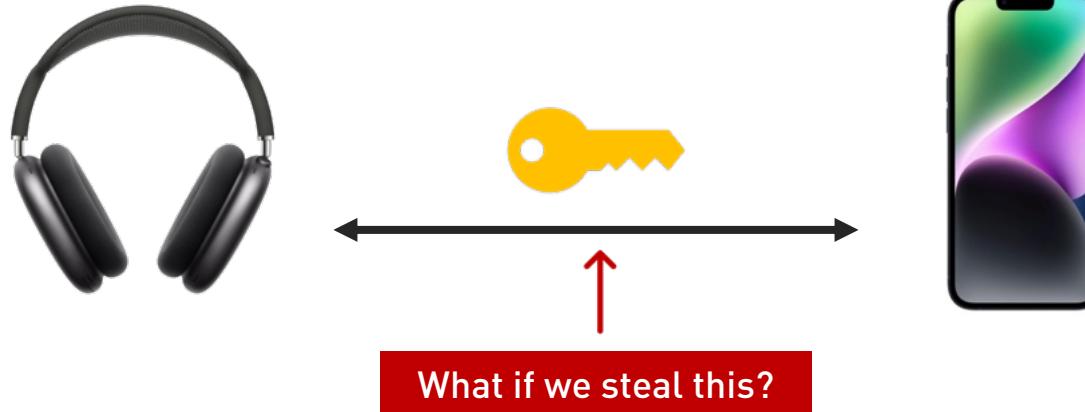
voice assistants

Hands-Free Profile (HfP)

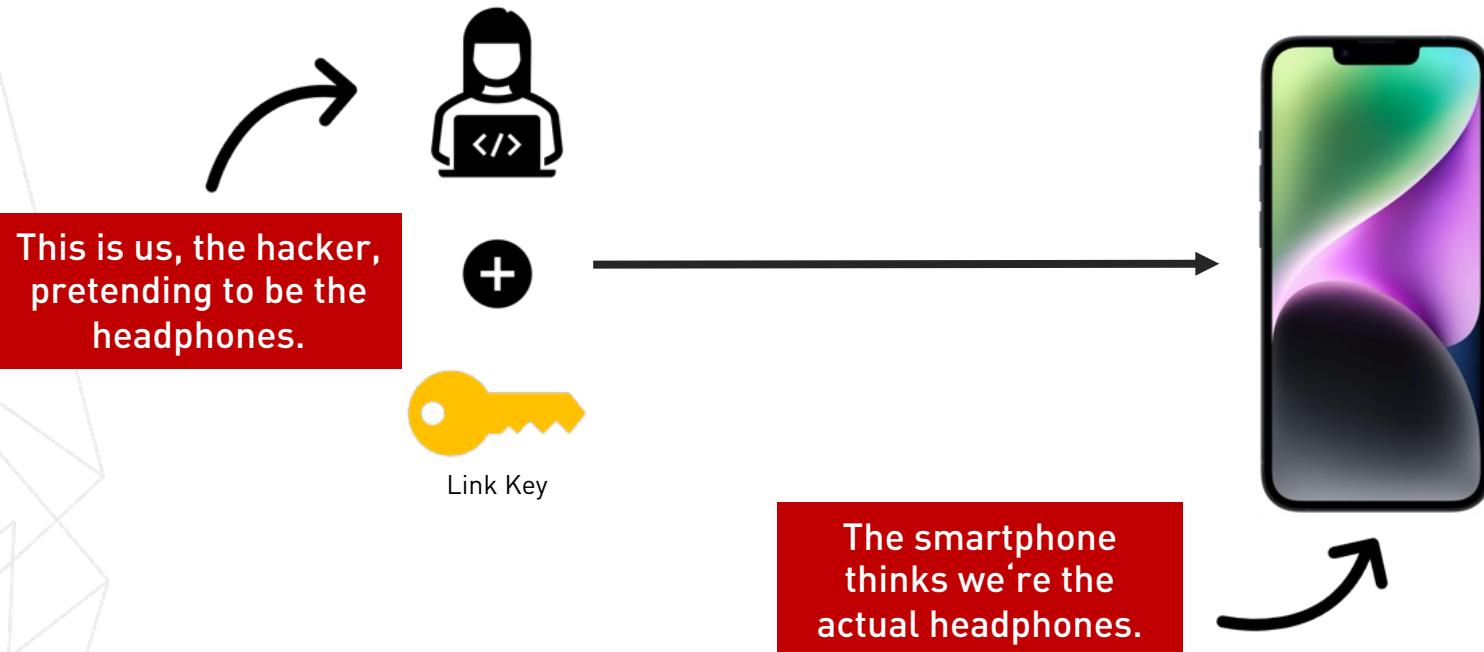
# Bluetooth Classic: Hands-Free Profile (HFP)

- **Call Control**
  - Answer, reject, end calls
  - Dial a number (including redialing the last number)
- **Audio Routing**
  - Transfer audio between phone and hands-free device
- **Caller Information Access**
  - Read caller ID
  - Access signal strength, battery level
- **Network Information**
  - Query carrier/operator name
  - Subscriber Number (local phone number)
- **Voice Assistant**
  - Trigger Voice Assistant (Gemini/Google Assistant/Siri)
- **Phone Book Access**
  - Non-standard AT commands for phonebook access
  - Requires permissions

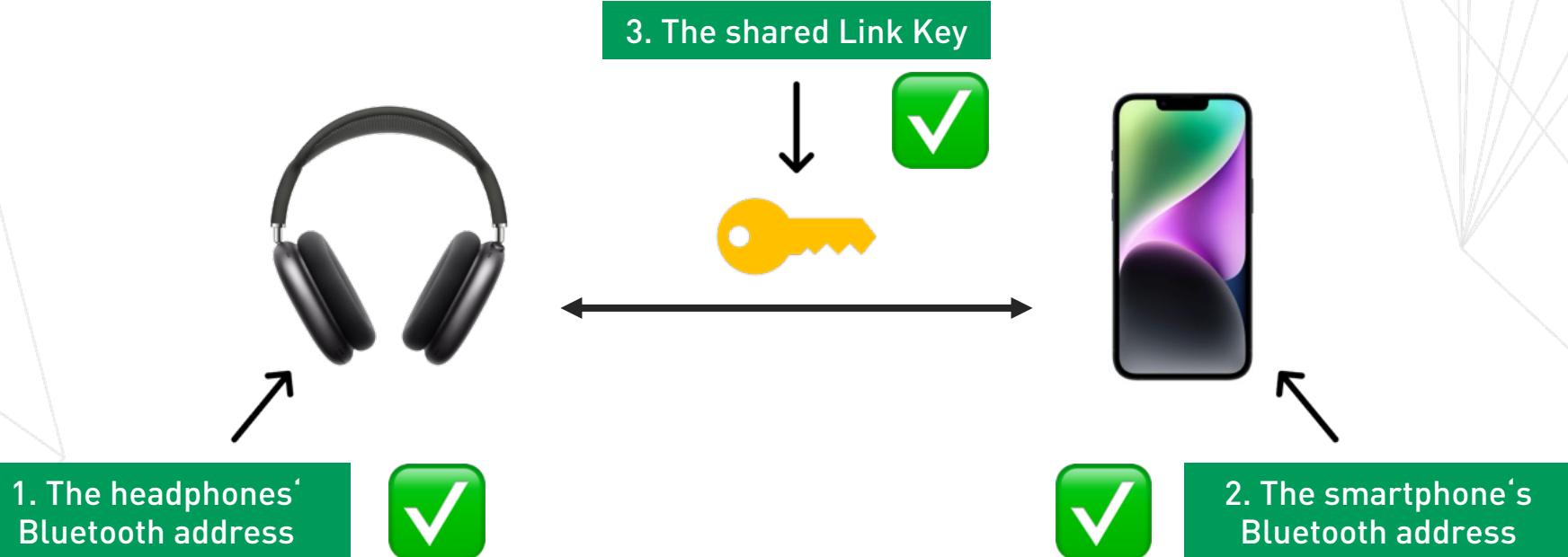
## Let's get back to the cryptographic bond



# Impersonating the Headphones to the Smartphone



# But what do we need for that?



# Live Demo

Sun 21. Dec

10:40



Amazon

2m ago

Amazon: 624348 is your sign-in code. For  
your security, don't share it. Amazon will  
never contact you to ask for this code.

WhatsApp

▼ Show less



Logged out of WhatsApp

3m ago

Your phone number is no longer  
registered on this phone. Tap to view.



WhatsApp

4m ago

WhatsApp registration code was  
requested for your phone number.



Code for new phone

5m ago

Do not share it with anyone. Your code  
is 420592.



<https://github.com/bluekitchen/btstack>

 **btstack** Public

master 3 Branches 21 Tags Go to file Add file Code

andreifominykh and mringwal	chipset/realtek: fix compile errors	4bdc72d · 3 months ago
.github/ISSUE_TEMPLATE	github: improve issue instructions	3 years ago
3rd-party	3rd-party/qr-code-generator: avoid warning if compiler c...	8 months ago
chipset	chipset/realtek: fix compile errors	3 months ago
doc	doc/manual: update tables layout	4 months ago
example	Merge branch 'develop'	4 months ago
platform	posix: use -b for baudrate and -m for BD_ADDR on comm...	5 months ago
port	libusb: use findpacket python3	4 months ago
src	broadcast_audio_uri_builder: fix warning, simplify boolea...	4 months ago
test	test/hfp: fix tests	4 months ago
tool	tool/misc: add script to move extra branches to btstack-e...	4 months ago
.gitignore	gitignore: add .DS_Store	7 months ago
CHANGELOG.md	Merge branch 'develop'	4 months ago
LICENSE	btstack: add licence to main folder	5 years ago
README.md	Merge branch 'develop'	4 months ago

[README](#) [License](#)

## Welcome to BTstack

BTstack is [BlueKitchen's](#) implementation of the official Bluetooth stack. It is well suited for small, resource-constraint devices such as 8 or 16 bit embedded systems as it is highly configurable and comes with an ultra small memory footprint.

### About

Dual-mode Bluetooth stack, with small memory footprint.

 [bluekitchen-gmbh.com](#)

-  Readme
-  View license
-  Activity
-  Custom properties
-  2k stars
-  124 watching
-  669 forks

[Report repository](#)

### Releases

 21 tags

### Packages

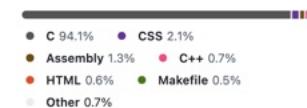
No packages published

### Contributors



[+ 29 contributors](#)

### Languages



## Shoutout to BTStack!

# Disclosure

# These are the devices we initially saw and were able to confirm

Bose
<ul style="list-style-type: none"><li>• Quiet Comfort Earbuds<sup>1,4</sup></li></ul>

Jabra
<ul style="list-style-type: none"><li>• Elite 8 Active<sup>1,3</sup></li></ul>

JBL
<ul style="list-style-type: none"><li>• Live Buds 3</li><li>• Endurance Race 2</li></ul>

Beyerdynamic
<ul style="list-style-type: none"><li>• Amiron 300</li></ul>

Jlab
<ul style="list-style-type: none"><li>• Epic Air Sport ANC</li></ul>

Teufel
<ul style="list-style-type: none"><li>• Airy TWS 2</li></ul>

Sony
<ul style="list-style-type: none"><li>• WH-1000XM6<sup>2</sup></li><li>• WF-1000XM3</li><li>• WF-1000XM5</li><li>• CH-720N</li><li>• WH-1000XM5<sup>2,3</sup></li><li>• WF-1000XM4</li><li>• WH-1000XM4<sup>2,3</sup></li><li>• WH-XB910N</li><li>• WI-C100</li><li>• WH-CH520</li><li>• ULT Wear</li><li>• WF-C510-GFP</li><li>• WF-C500</li><li>• Link Buds S</li></ul>

Marshall
<ul style="list-style-type: none"><li>• WOBURN III</li><li>• STANMORE III</li><li>• ACTON III</li><li>• MAJOR V</li><li>• MINOR IV</li><li>• MOTIF II</li></ul>

Things we noticed (not exhaustive):

- 1: No BLE (GATT)
- 2: Some protocol messages are filtered
- 3: Classic pairing is required
- 4: Classic pairing is sometimes(?) required

# This is our current assumption

Bose
• Quiet Comfort Earbuds
MoerLabs
• EchoBeatz

Jabra
• Elite 8 Active
Jlab
• Epic Air Sport ANC

JBL
• Live Buds 3 • Endurance Race 2

Beyerdynamic
• Amiron 300
Teufel
• Airy TWS 2

Sony
• WH-1000XM6 • WF-1000XM3 • WF-1000XM5 • CH-720N • WH-1000XM5 • WF-1000XM4 • WH-1000XM4 • WH-XB910N
• WI-C100 • WH-CH520 • ULT Wear • WF-C510-GFP • WF-C500 • Link Buds S

Marshall
• WOBURN III • STANMORE III • ACTON III • MAJOR V • MINOR IV • MOTIF II

- Some vendors have released updates since the disclosure.
- Only some of them explicitly mentioned fixing the CVEs.
- Some put: “Fix a security vulnerability” or “Stability and performance improvement”
- We might be missing patches! This is only *our* current overview.
- We don’t have the time and resources to check and monitor updates.

Airoha

(SoC, SDK, reference design)

Vendor 1

Vendor 2

OEM/ODM 1

Vendor x

Product 1

Product 2

Product 3

Product 1

Product 2

Product 3

Product 4

Vendor n

Vendor n+1

Product 1

Product 1

Product 2

Product 1

Product 2

Product 3

Consumer

Airoha

(SoC, SDK, reference design)

We can search for  
these, but we will  
never find them all.

Vendor 1

Vendor 2

OEM/ODM 1

Vendor x

Product 1

Product 2

Product 1

Product 2

Product 3

Product 4

Vendor n

Vendor n+1

Product 1

Product 2

Product 3

Product 1

Product 1

Product 2

Consumer

We might be able to identify and talk to a few of them.

Airoha

(SoC, SDK, reference design)

Vendor 1

Vendor 2

OEM/ODM 1

Vendor x

Product 1

Product 2

Product 3

Product 1

Product 2

Product 3

Product 4

Product 5

Vendor n

Vendor n+1

Product 1

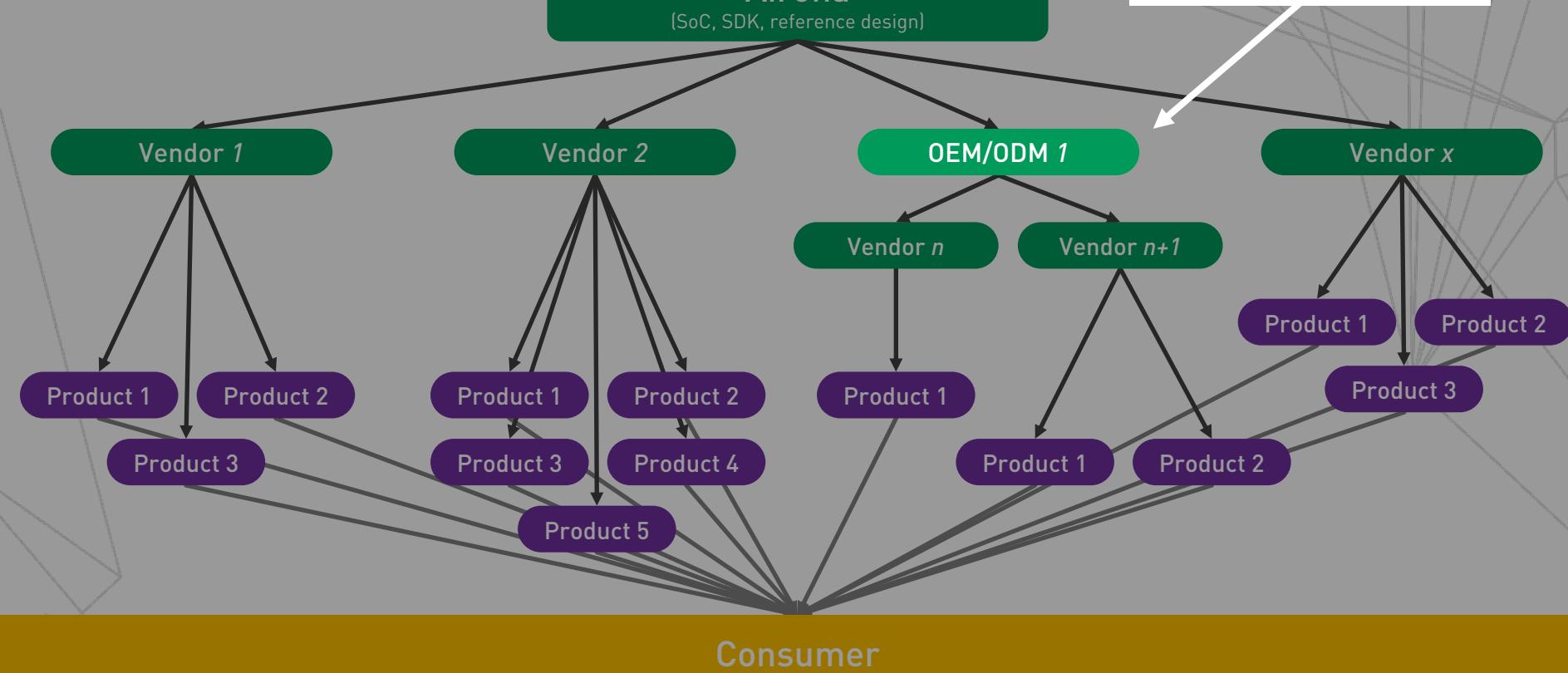
Product 2

Product 3

Consumer

OEMs will also be difficult to identify.

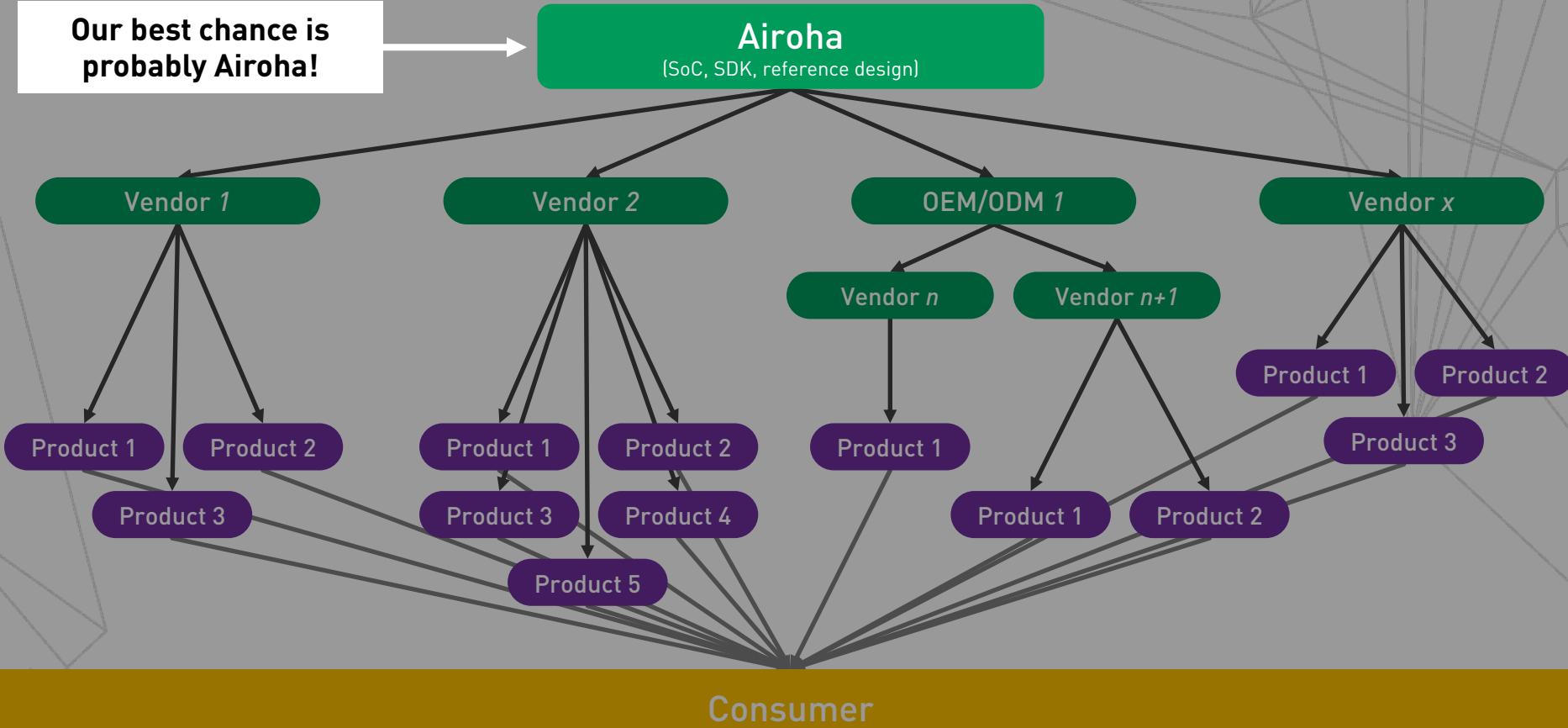
Airoha  
(SoC, SDK, reference design)



**Our best chance is  
probably Airoha!**

Airoha

(SoC, SDK, reference design)



# Airoha's security disclosure page looks really good!

**AIROHA** ≡

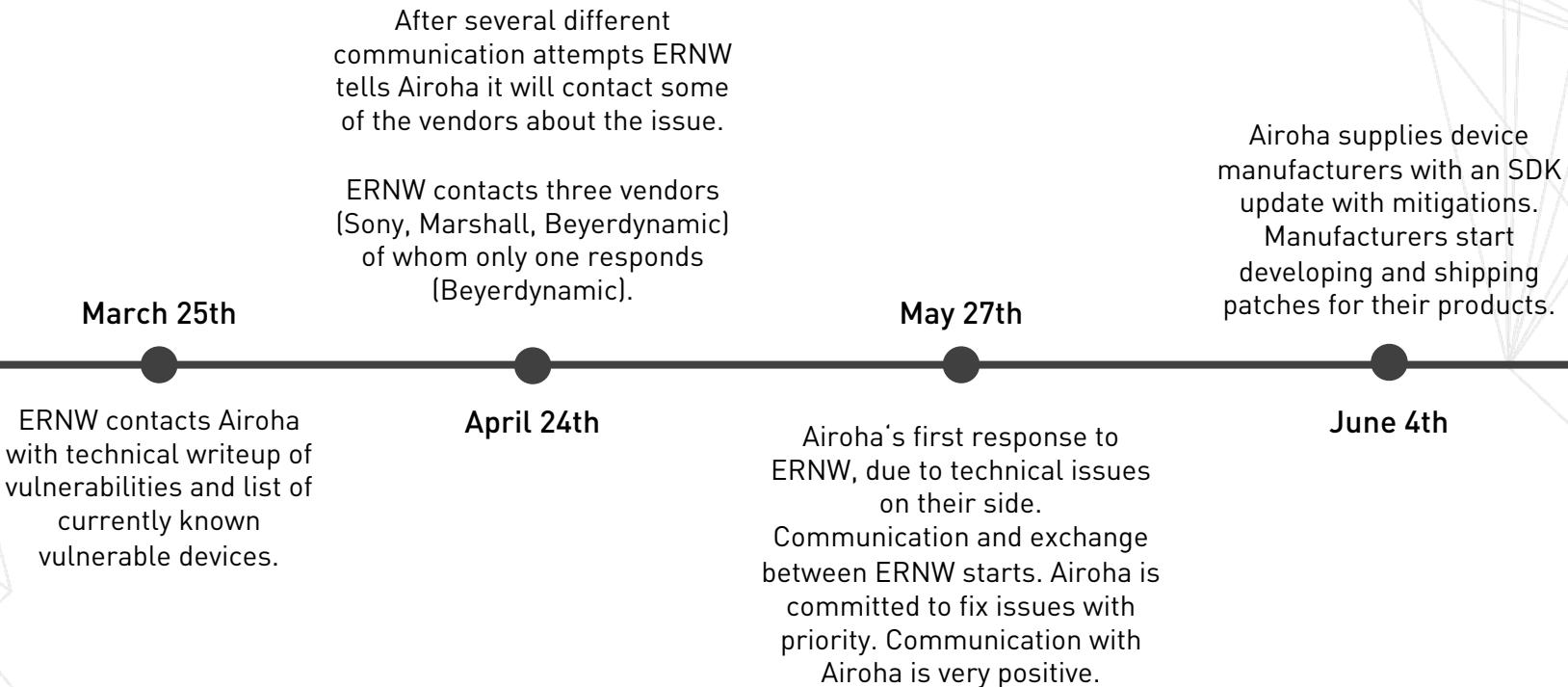
## What You Can Expect From Us

- We aim to respond within a maximum of 3 to 5 business days upon receiving the initial report. If you do not hear back from us after a week you submitted the initial report, please send it to us again.
- We will make best effort to address the security vulnerabilities by including but not limited to releasing patches to our OEM partners within 90 days and communicate with the stakeholders as needed.

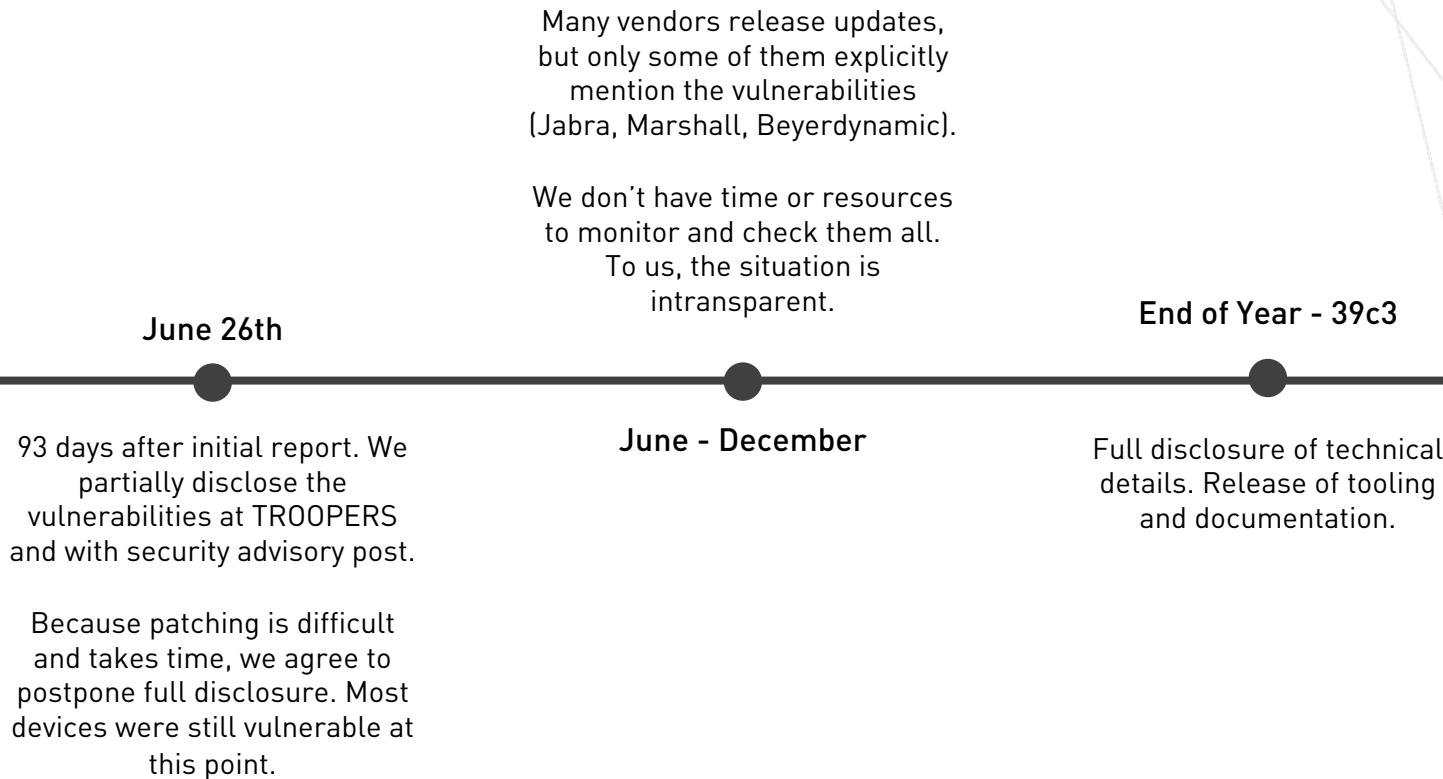
### • Can I use the encrypted channel to submit a security report?

Yes, please use [our PGP Public Key](#) to send the encrypted security report to [security@airoha.com](mailto:security@airoha.com).

# Disclosure Timeline: Short Version



# Disclosure Timeline: Short Version



<https://github.com/auracast-research/race-toolkit>



## Goals:

- Enable users to check their devices.
- No weaponized tooling.
- Tooling and documentation for further research.

### README

## RACE Toolkit

RACE Toolkit is the tool released alongside our Airoha research. You can find more about that in our [blog post](#).

This repository contains a Python-based command-line toolkit for interacting with devices that expose the **RACE protocol** over various transports (BLE GATT, Bluetooth Classic RFCOMM, USB HID). It is primarily intended for further security research into the Airoha ecosystem and for end-users to check whether their devices are affected by the vulnerabilities.

### About

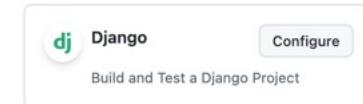
Communicate with Airoha-based Bluetooth devices using the RACE protocol via Bluetooth and USB.

### Packages

No packages published  
[Publish your first package](#)

### Suggested workflows

Based on your tech stack



# FOTA – Firmware Updates

- We reimplemented the firmware update mechanism.
  - flash valid, "signed" firmware files
  - Sony: MDR Proxy repository ([https://github.com/lzghzr/MDR\\_Proxy](https://github.com/lzghzr/MDR_Proxy))
- It's also possible to downgrade firmware.
  - Only for research, for your personal device want to have the newest FW with the vulns patched.
- Unpack and pack the firmware to potentially manipulate it
  - ! during that process we bricked two devices, so we currently don't recommend trying that.
- Currently only works for headphones, not for True Wireless earbuds.

## 6.1.1 nRF Connect Mobile on iOS

### Step 1: Filtering

The first thing that should be done is setting a filter. There are usually many Bluetooth devices in the vicinity. Tapping the arrow button on the top right opens up the filter menu. Switch on *Remove Unnamed* and *Remove Unconnectable*. We just want to show devices that are actually connectable, and we want to be able to identify them by their name. If the target device was not renamed explicitly, it should be advertised under its model name (e.g. WH-CH720N in this example).

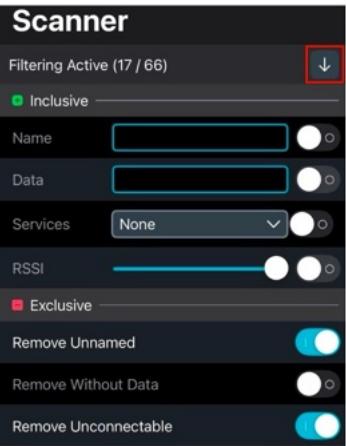


Figure 17: iOS Set Filter

### Step 2: Connecting

Once the filter is active, you can look for the target device. In this example, we're connecting to the Sony WH-CH720N headphones. Tap the *Connect* button to establish a BLE connection. Some device models might request pairing at this point. Accept the pairing if necessary. If there's no prompt for a pin or button press, this is still not a proper authentication.

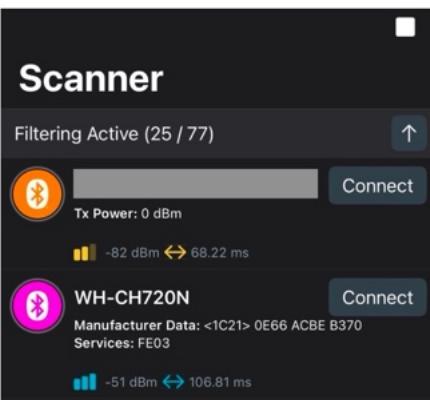


Figure 18: iOS Connect to Device

## Blog Post/White Paper:

Check if your device is affected via BLE using the Nordic nRF Connect Mobile App. This is a low-barrier method to quickly check a device.

# A quick note on mitigation

Mitigation	Manufacturers	Users
<ul style="list-style-type: none"><li>• Enforce authentication for Bluetooth Classic connections (CVE-2025-20701).</li><li>• Enforce authentication for sensitive GATT services (CVE-2025-20700).</li><li>• Don't expose critical debug-like interfaces via Bluetooth (CVE-2025-20702).</li></ul>	<ul style="list-style-type: none"><li>• Do a security assessment before you sell your devices.</li><li>• E.g. BSAM: Bluetooth security assessment methodology. This would have caught at least the pairing issue.</li><li>• Offer security contacts in addition to bug bounty platforms.</li></ul>	<ul style="list-style-type: none"><li>• Update your headphones!</li><li>• If your vendor does not mention the CVEs, check your device with race-toolkit.</li><li>• Maybe use cable headphones if you feel like you might be a target.</li><li>• Remove old Bluetooth pairings.</li></ul>

## Closing Thoughts

- As smartphones are getting more secure, peripherals become more interesting for attackers.
- Even physical compromise of peripherals may be worth the effort if it allows access to the phone.





GitHub: RACE Toolkit



- Check the blog post & white paper for more technical details and how to check your device.
- Come find us at 39c3 if you have questions about the tooling, your headphones or think you found another affected device!
- Links:
  - Blogpost: <https://insinuator.net/2025/12/bluetooth-headphone-jacking-full-disclosure-of-airoha-race-vulnerabilities>
  - RACE Toolkit: <https://github.com/auracast-research/race-toolkit>
  - Whitepaper: <https://ernw.de/en/publications.html>



dheinze at ernw.de  
fsteinmetz at ernw.de



ttdennis@chaos.social  
twillnix@infosec.exchange



**ERNW**  
providing security.