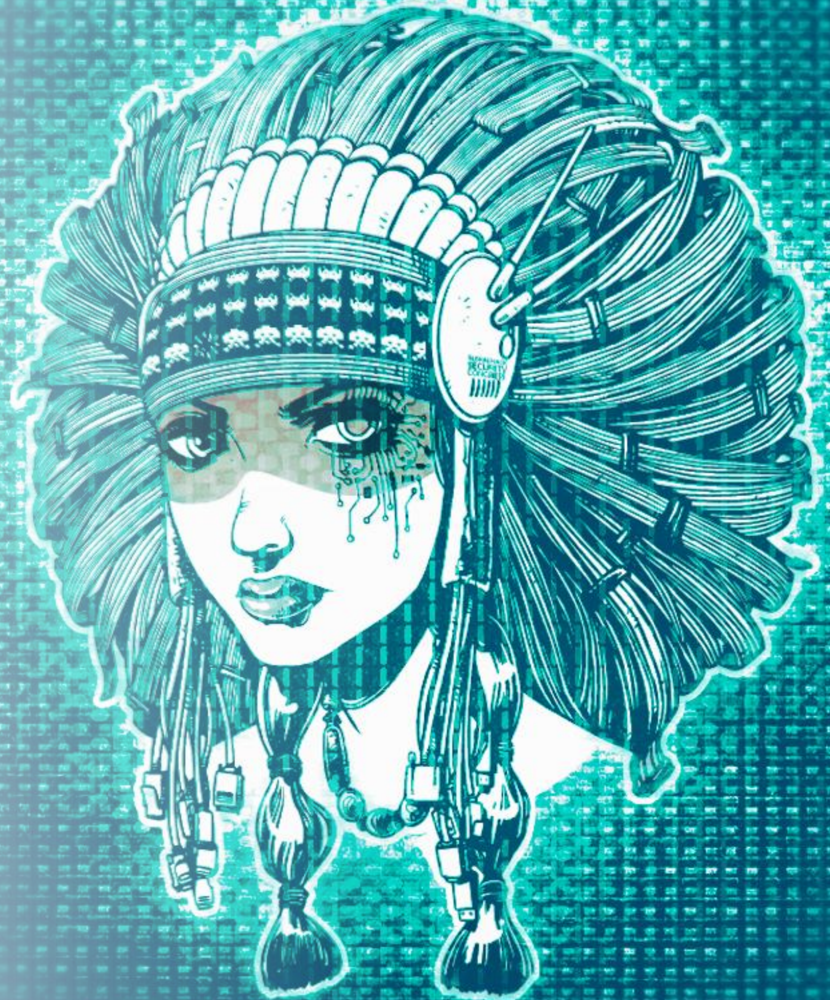
EuskalHack Security Congress VI

LOS ENTRESIJOS DE BLUETRUST

TARLOGIC
CYBERSECURITY EXPERTS



$ WHOAMI

# Jesús M. Gómez Moreno

#Research Engineer at Tarlogic Security

@zus_999

@zus@masto.es

jesus.gomez@tarlogic.com

$ WHOAMI

# Antonio Vázquez Blanco

#Research Engineer at Tarlogic Security

@antoniovazquezblanco@mastodon.social

antonio.vazquez@tarlogic.com

ÍNDICE

01 | STATE OF THE ART

- BIAS
- KNOB
- BIAS+KNOB

02 | CONTRIBUTIONS

- BLUETRUST
- TOOL CONTRIBUTIONS

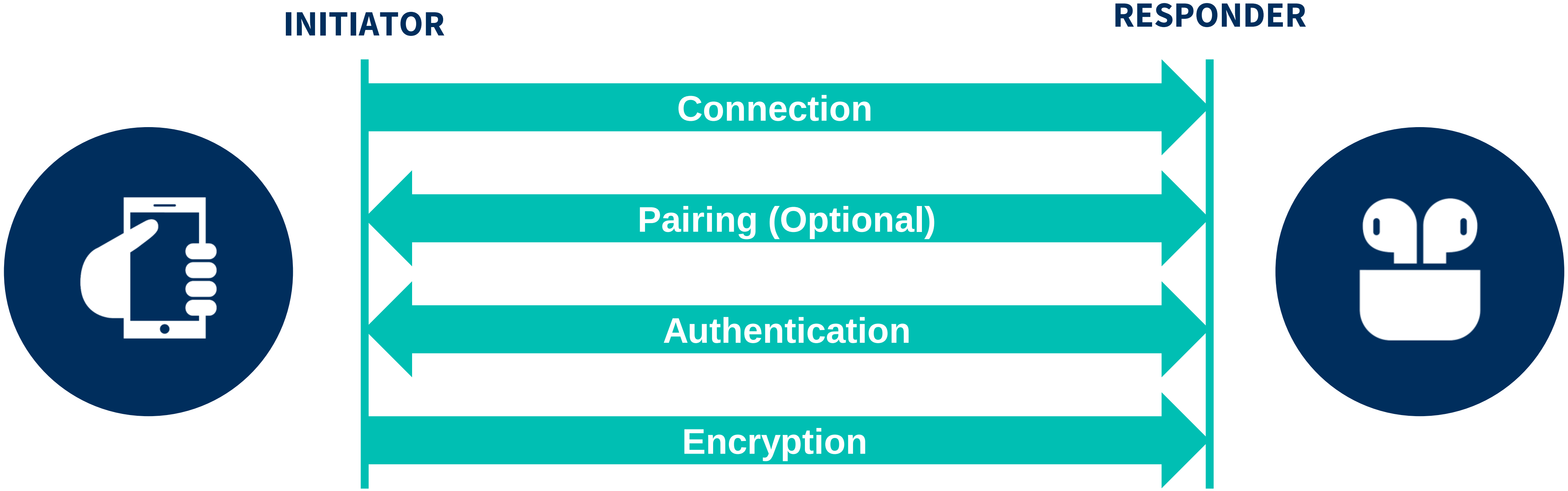
BIAS

Bluetooth Impersonation AttackS

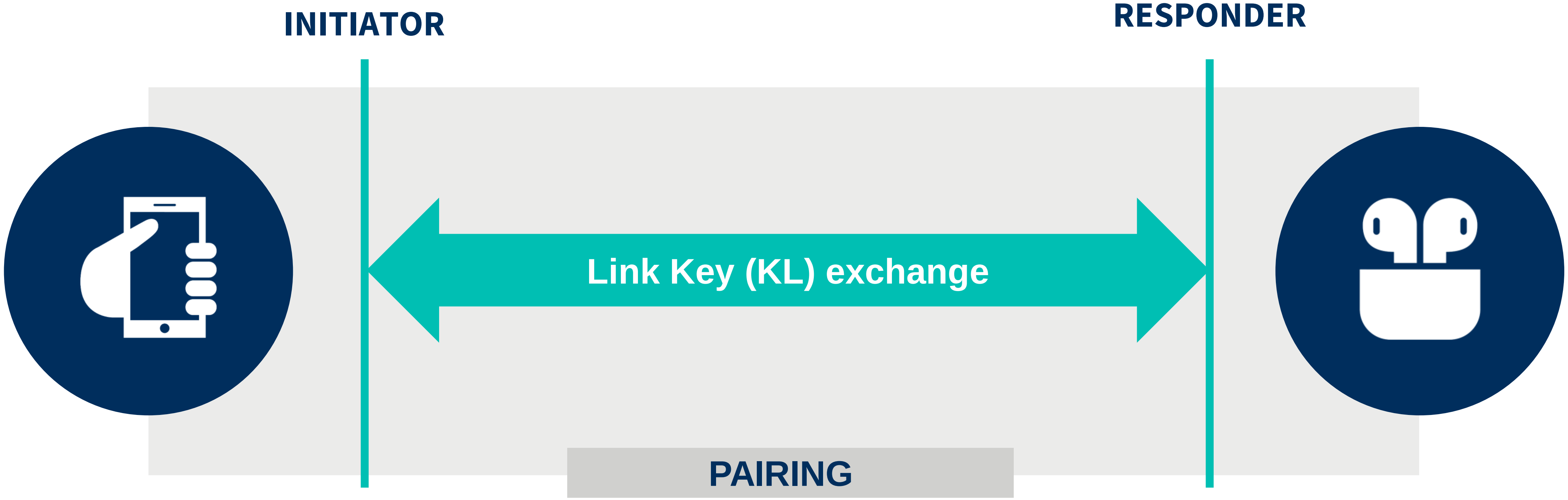
[CVE 2020-10135]

(Theory)

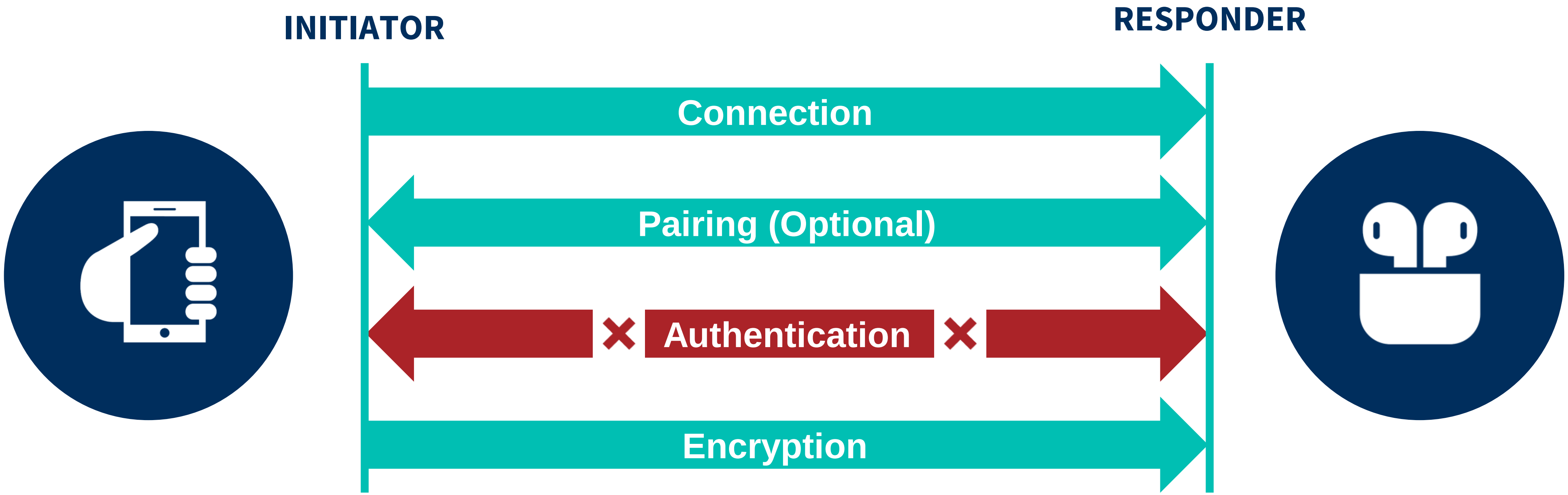
BIAS A COMMON BLUETOOTH CONNECTION



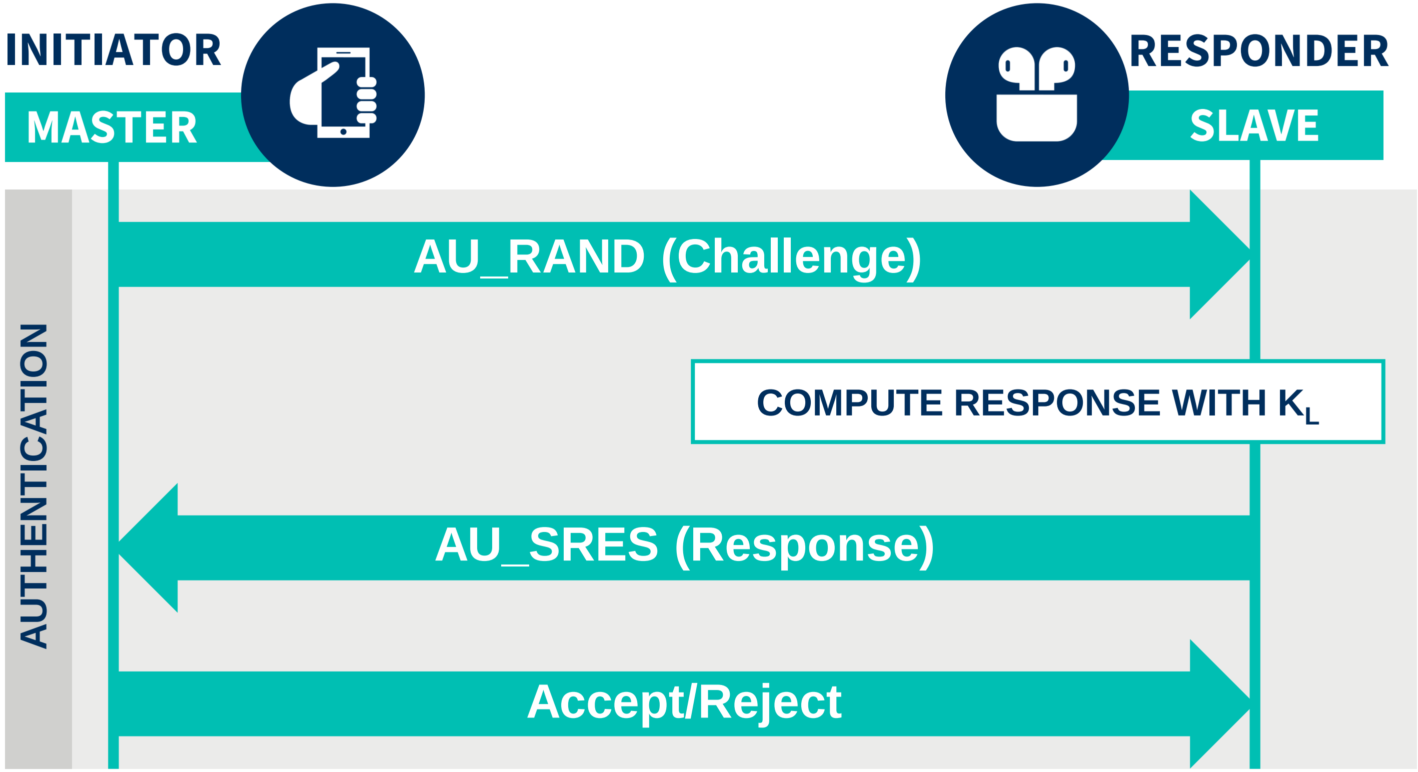
BIAS PAIRING



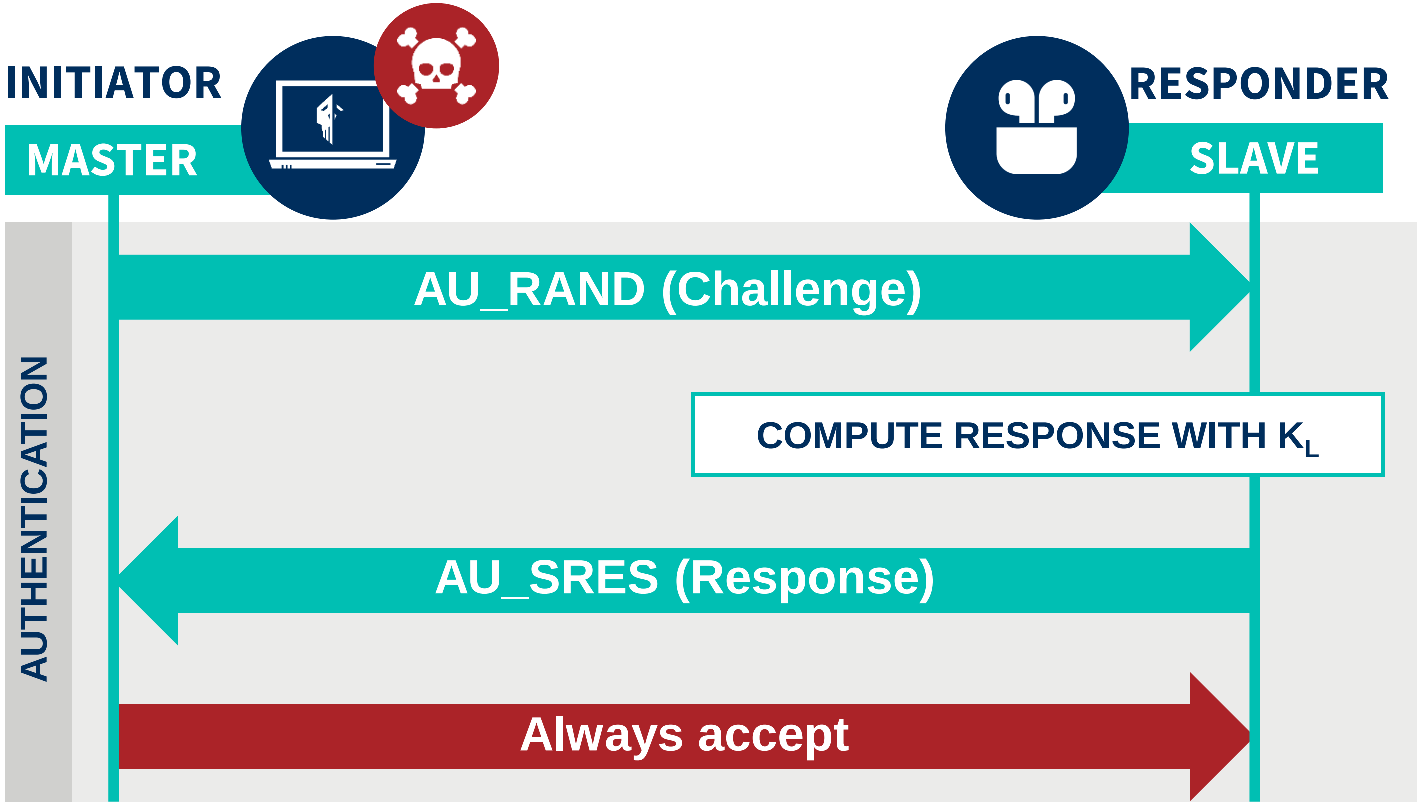
BIAS WHAT IS IT?



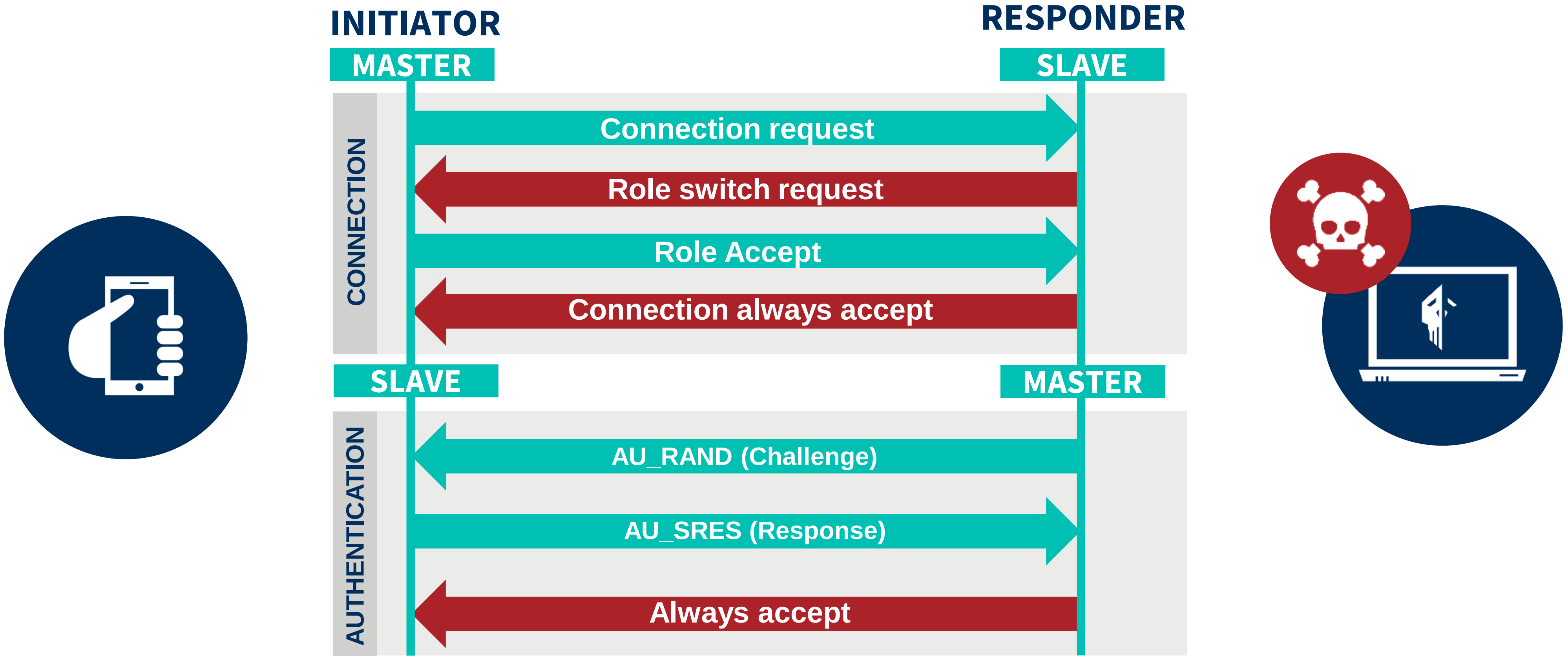
BIAS LEGACY AUTHENTICATION



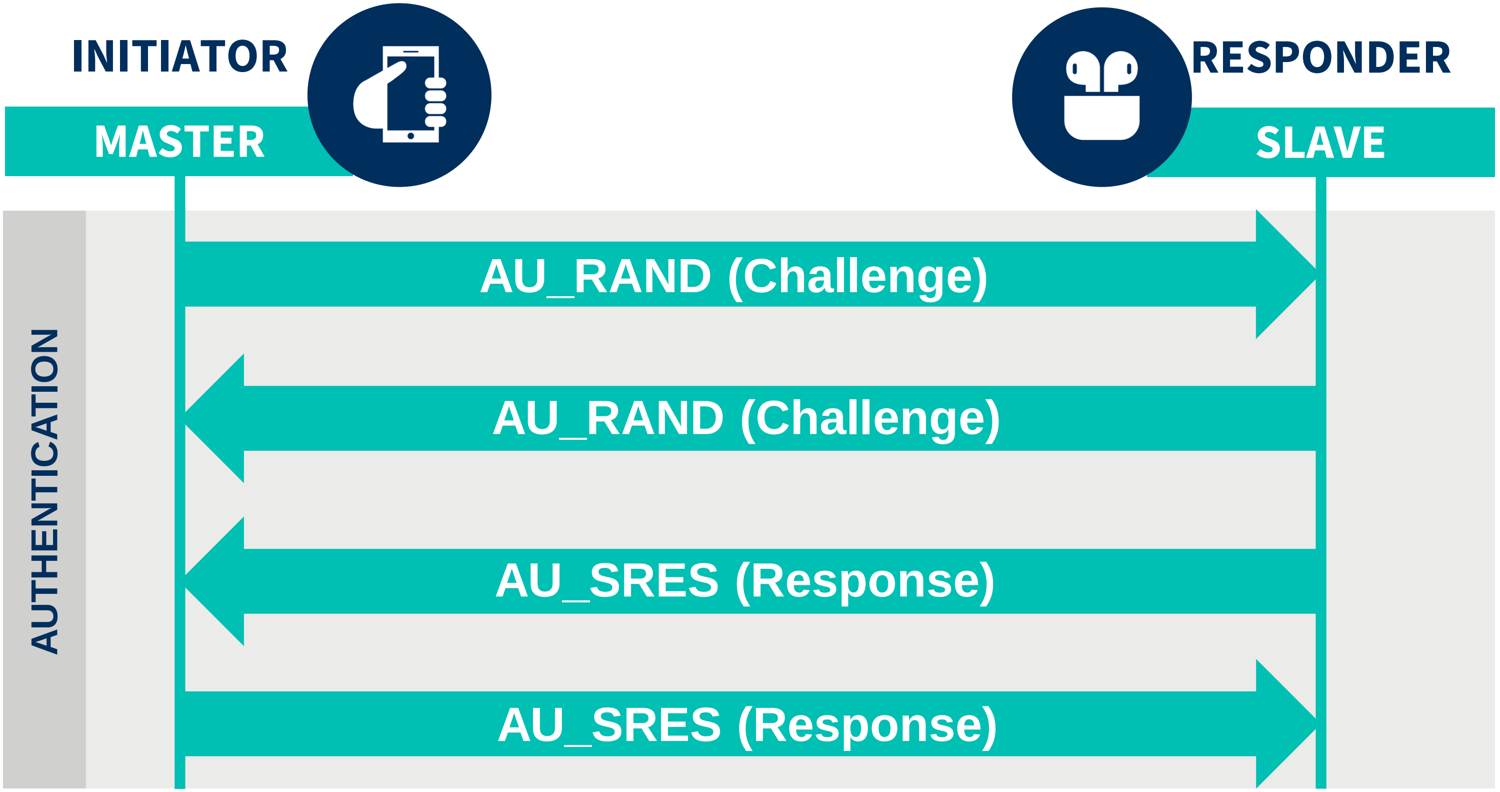
BIAS LEGACY AUTHENTICATION PARTIAL BYPASS



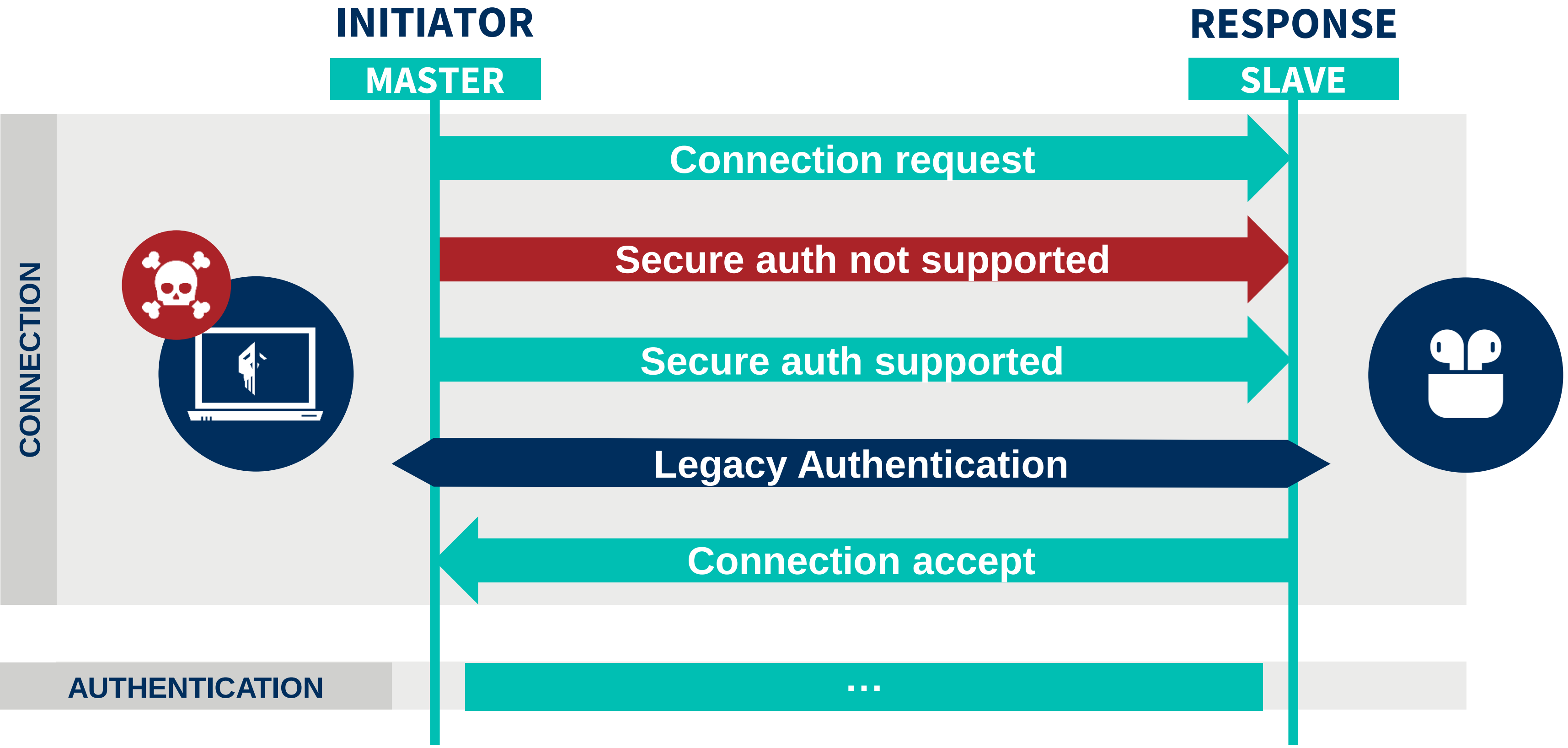
BIAS LEGACY AUTHENTICATION TOTAL BYPASS



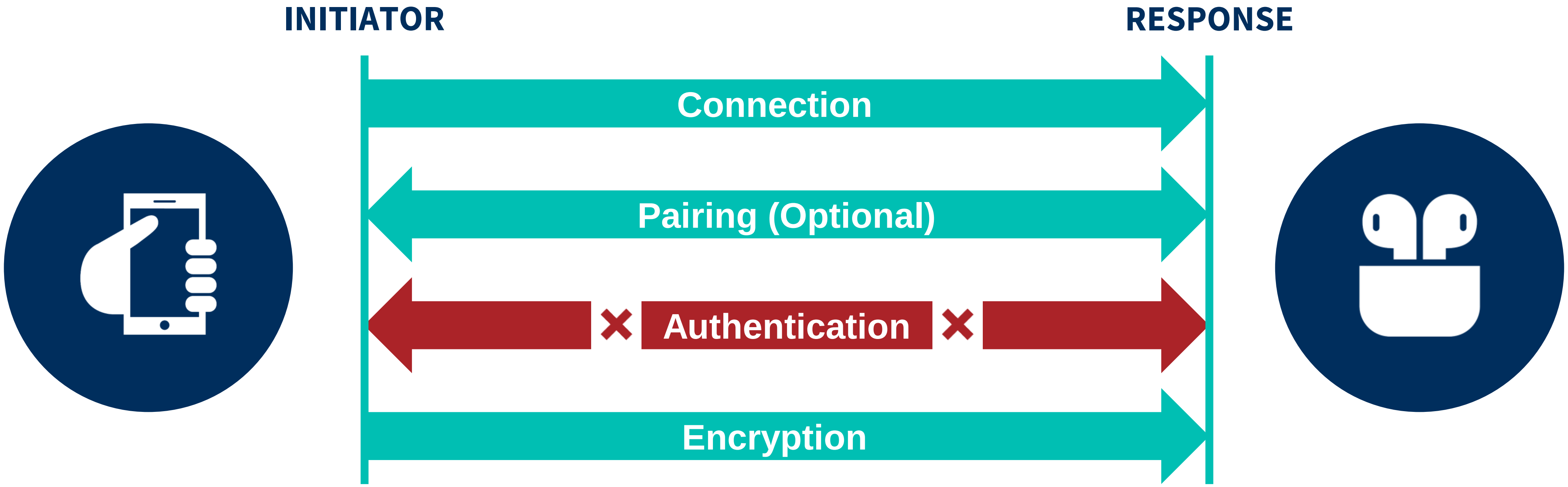
BIAS SECURE AUTHENTICATION



BIAS SECURE AUTHENTICATION DOWNGRADE



BIAS WHERE ARE WE?



BIAS MITIGATIONS

None known

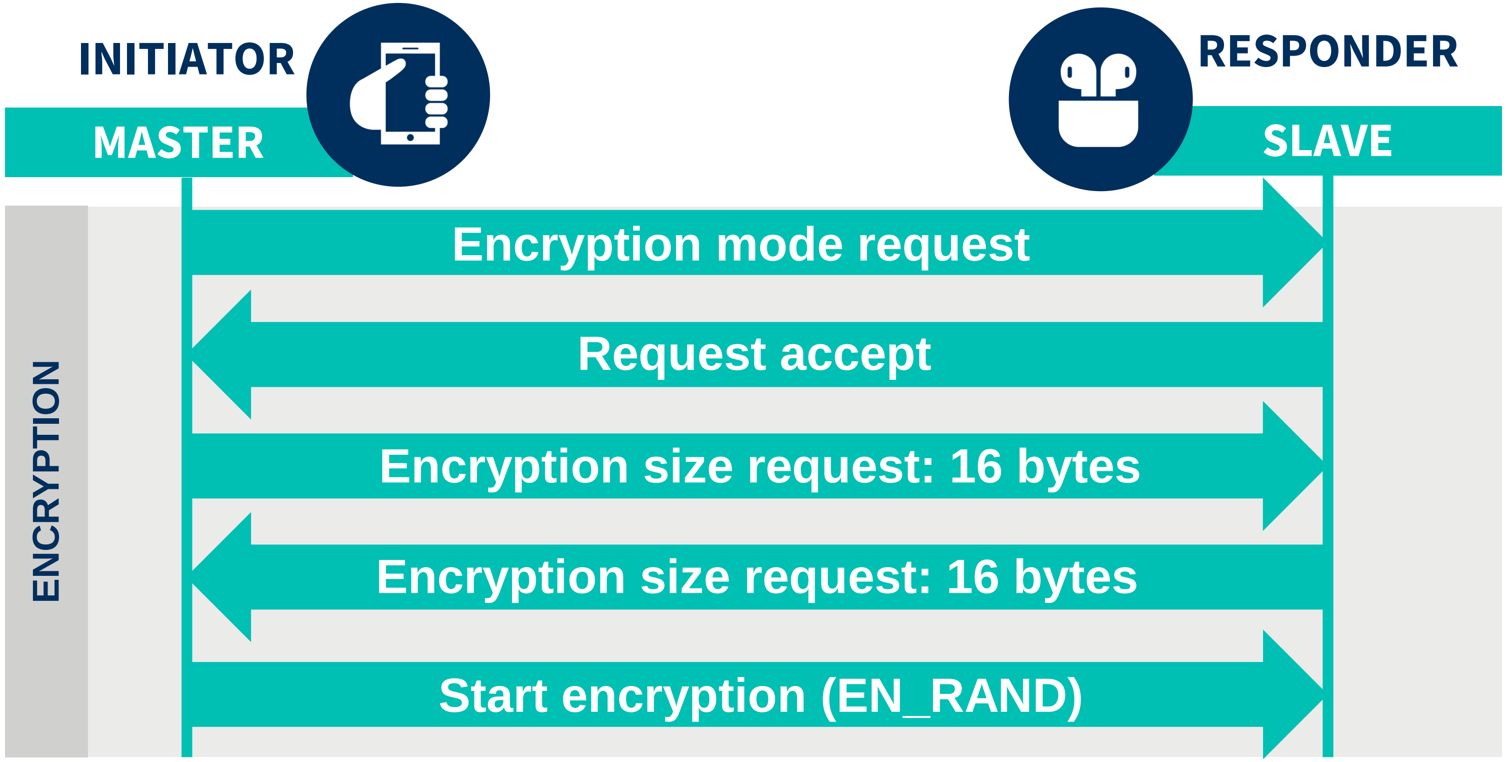
KNOB

Key Negotiation Of Bluetooth

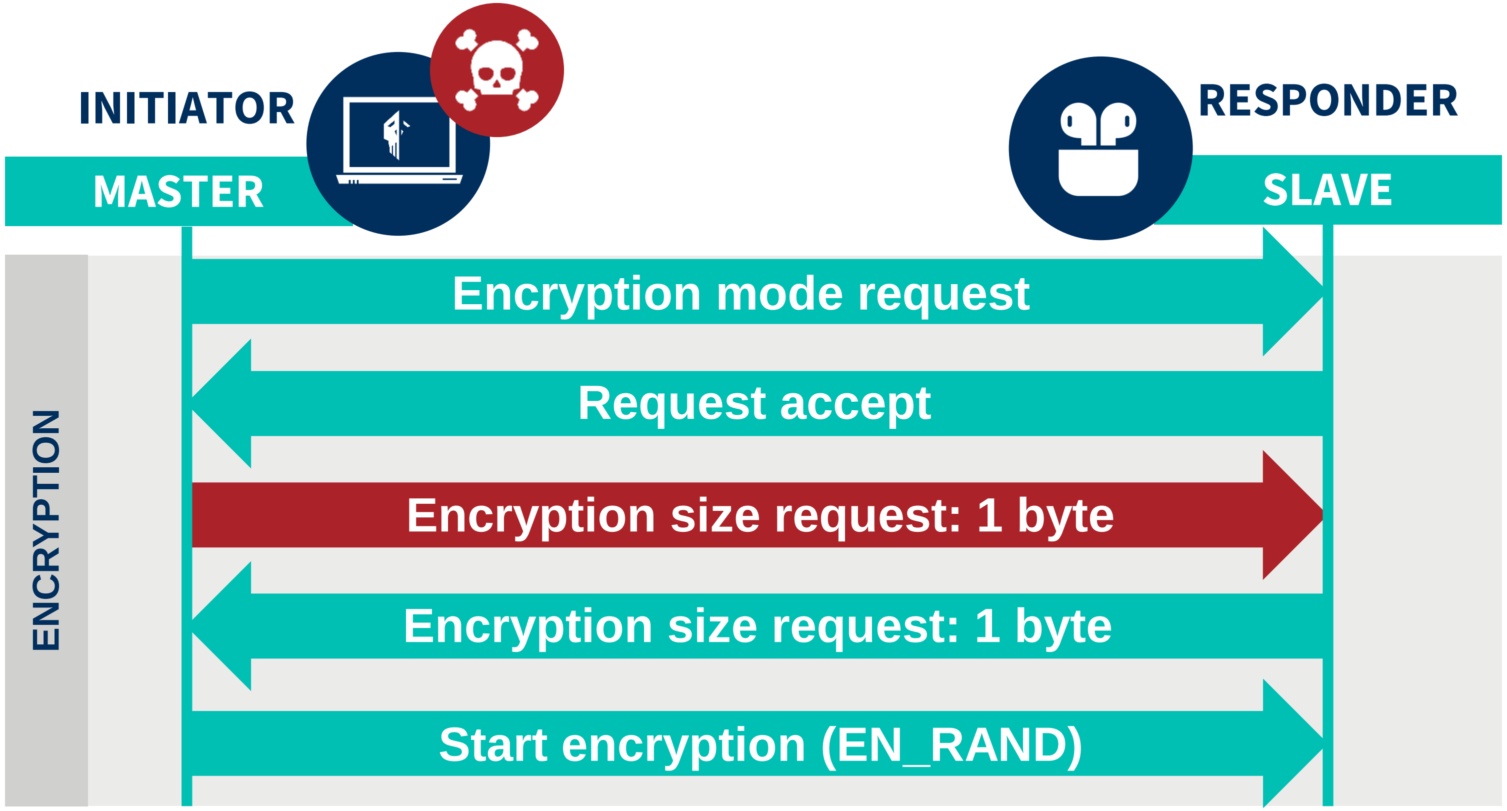
[CVE 2019-9506]

(Theory)

KNOB NORMAL ENCRYPTION PROCESS


INITIATOR
MASTER
RESPONDER
SLAVE
ENCRYPTION
Encryption mode request
Request accept
Encryption size request: 16 bytes
Encryption size request: 16 bytes
Start encryption (EN_RAND)

KNOB THE ATTACK



INITIATOR
MASTER
RESPONDER
SLAVE
ENCRYPTION
Encryption mode request
Request accept
Encryption size request: 1 byte
Encryption size request: 1 byte
Start encryption (EN_RAND)

KNOB

MITIGATIONS

- Bluetooth LE stablishes a **MINIMUM KEY ENTROPY OF 7 BYTES.**

- Some devices **WILL REFUSE TO ACCEPT LOW ENTROPY KEYS (manufacturer dependant)**

BIAS & KNOB

(Practice)

BIAS & KNOB

THE ~~PROBLEMS~~ CHALLENGES

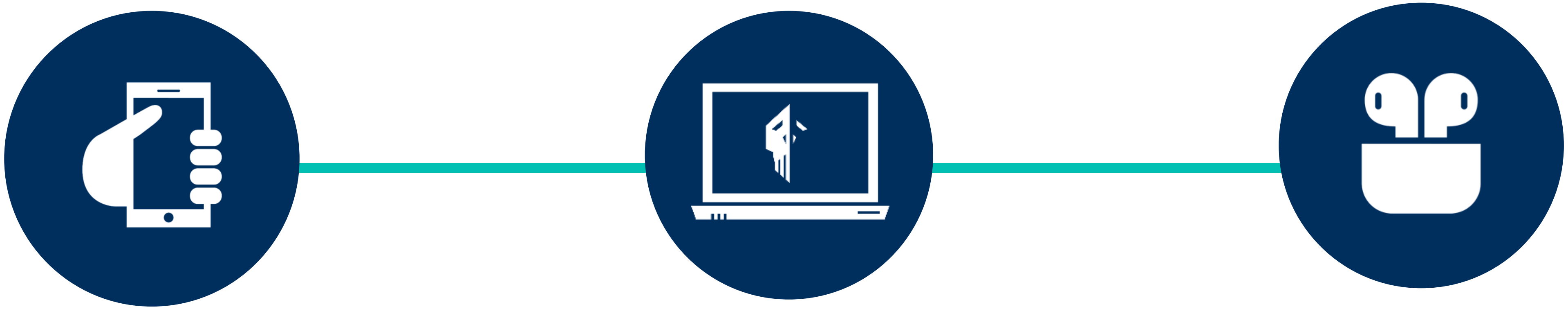
- Bluetooth is a complex standard. **Not every manufacturer follows it to the required detail** for the attacks to work…

- Attacks take place at the **lowest layers of Bluetooth**. Implementation requires tampering with **firmware of devices**...

- KNOB is viable in theory **but not in practice…**

KNOB

THE REAL PROBLEMS



ALICE

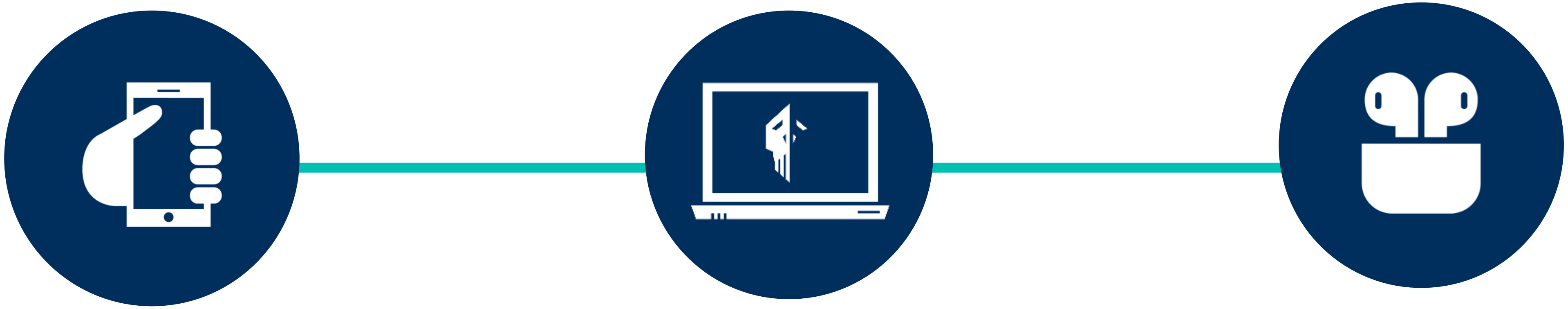
CHARLIE

ALICE´S HEADPHONES

KNOB

THE REAL PROBLEMS

Situation after Auth bypass (BIAS)



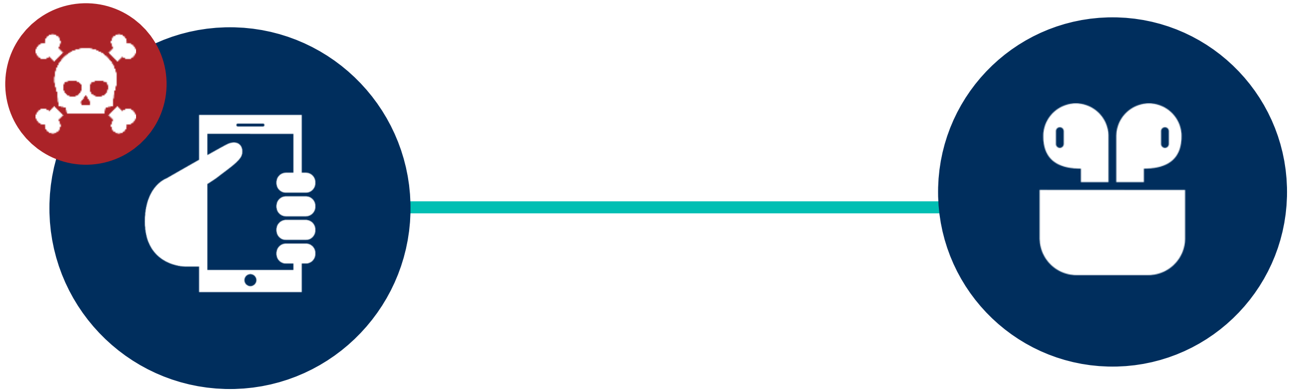
ALICE
(Slave)

CHARLIE
(Master on both connections)

ALICE´S HEADPHONES
(Slave)

KNOB

THE REAL PROBLEMS



PATCHED “ALICE”

ALICE´S HEADPHONES

BIAS & KNOB

CONCLUSIONS

- **Inmense gap** between theoretical and practical attacks…

- Papers are really good for theory but **often lack implementation details**…

- **PoCs** are useful but designed for specific cases that **do not reflect reality**…

OVERCOMING BIAS & KNOB

(Challenges)

OVERCOMING CHALLENGES

AUTOMATIC BTATTACH

Challenges

- Linux does not automatically recognize **Cypress CYW920819** as a Bluetooth device

- **Btattach** must be used to **let BlueZ recognize it**

OVERCOMING CHALLENGES

AUTOMATIC BTATTACH

Challenges

- Linux does not automatically **recognize Cypress CYW920819 as a Bluetooth device**
- **Btattach** must be used to let **BlueZ** recognize it

Fix

- Device recognition via **udev rules**
- Systemd service script that is triggered from the **udev rule** for the **btattach long running process**

OVERCOMING CHALLENGES

WIRESHARK HCI BROADCOM VENDOR DISSECTOR

Challenges

- **Debug** anything we have implemented…

- Access to **lower-level protocol messages** can only be done via **Vendor Proprietary protocols**

- There are no tools to inspect **Broadcom Vendor messages**

OVERCOMING CHALLENGES

WIRESHARK HCI BROADCOM VENDOR DISSECTOR

Challenges

- **Debug** anything we have implemented…
- Access to lower-level protocol messages can only be done via **Vendor Proprietary protocols**
- There are no tools to inspect **Broadcom Vendor messages**

Fix

- A **Wireshark HCI** Broadcom Vendor Dissector
- Reuse **lower-level protocol dissectors!**

OVERCOMING CHALLENGES

SCAPY SOCKETS

Challenges

- Programmatically interacting with our **Bluetooth adapter**

- **Fast prototyping of software** that interacts with Bluetooth low level protocols

- **Avoid BlueZ interfering** with our programs

- Access **all packets** from our adapter

OVERCOMING CHALLENGES

SCAPY SOCKETS

Challenges

- Programmatically interacting with our **Bluetooth adapter**
- **Fast prototyping of software** that interacts with Bluetooth low level protocols
- **Avoid BlueZ** interfering with our programs
- Access **all packets** from our adapter

Fix

- Implement **Bluetooth Monitor Channel Sockets and Bluetooth User Sockets in Scapy!** (Linux only)

OVERCOMING CHALLENGES

HCI SCAPY DISSECTORS

Challenges

- Programmatically interacting with our **Bluetooth adapter**
- **Fast prototyping of software** that interacts with Bluetooth at HCI level

OVERCOMING CHALLENGES

HCI SCAPY DISSECTORS

Challenges

- Programmatically interacting with our **Bluetooth adapter**
- **Fast prototyping of software** that interacts with Bluetooth at HCI level

Fix

- **Scapy HCI dissectors!**

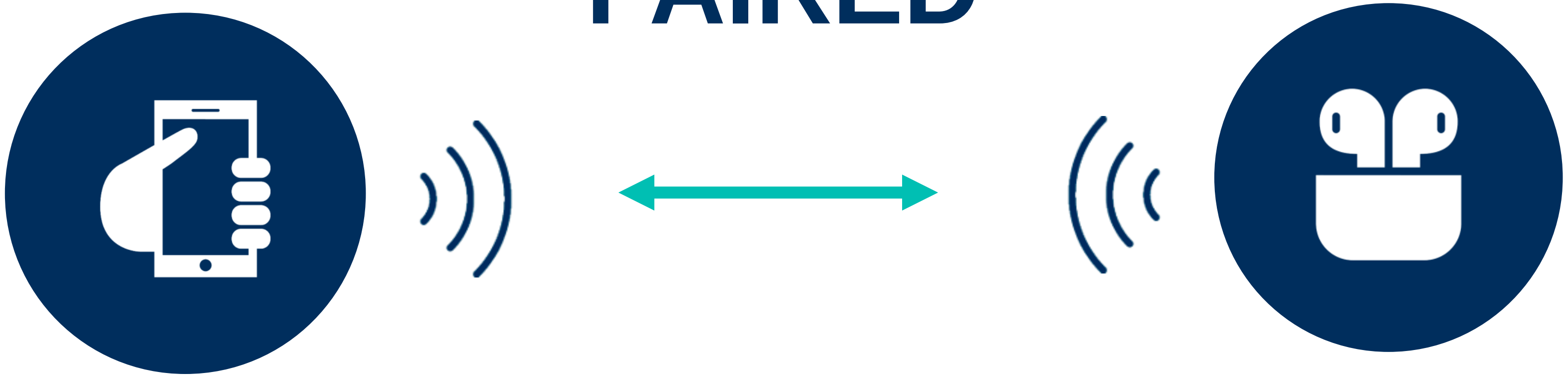
Now we know

BLUETRUST

Beyond BIAS & KNOB

BLUETRUST

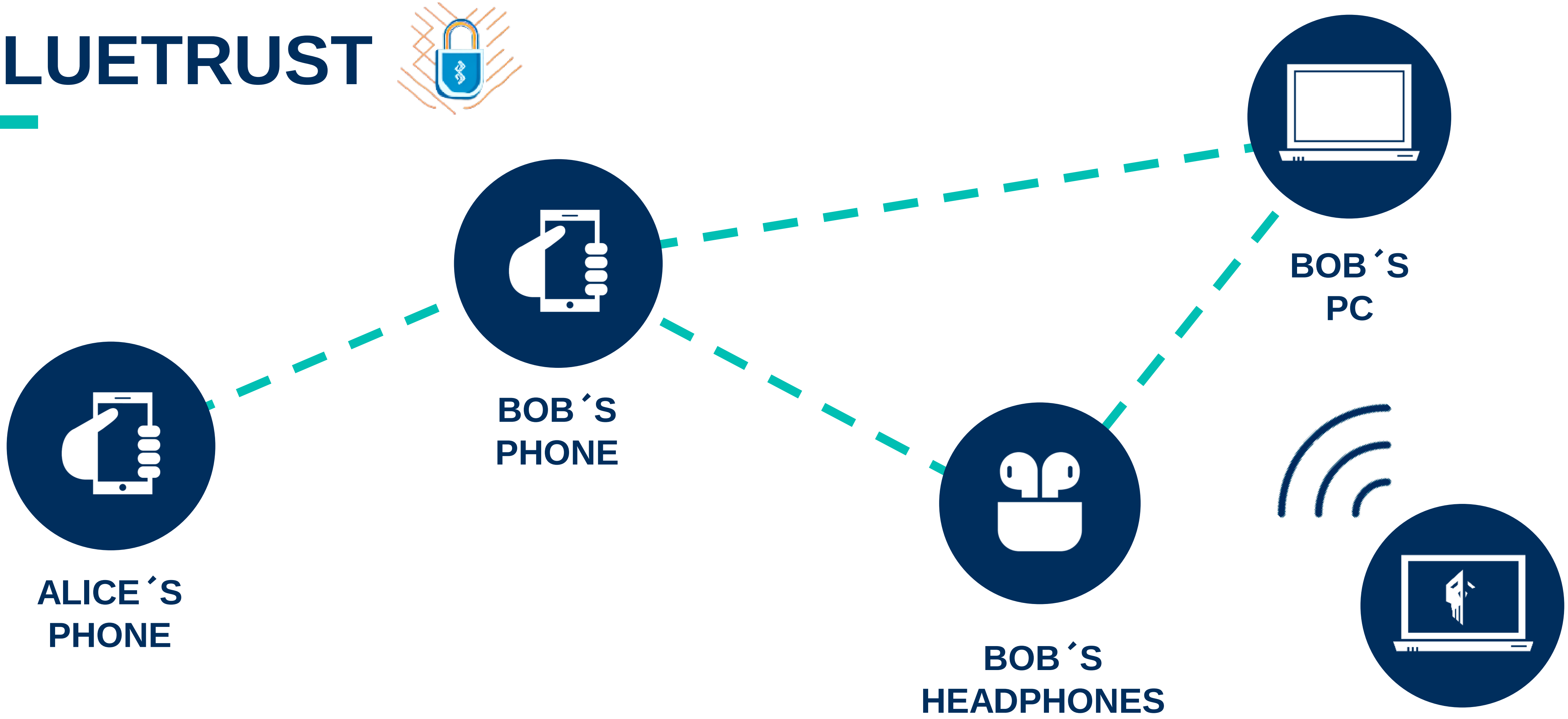
PAIRED



ALICE´S
PHONE

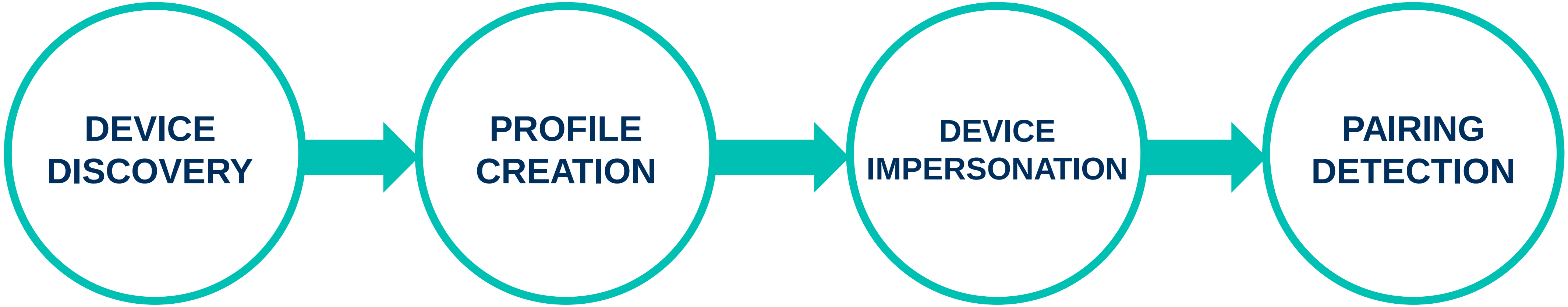
ALICE´S
HEADPHONES

BLUETRUST



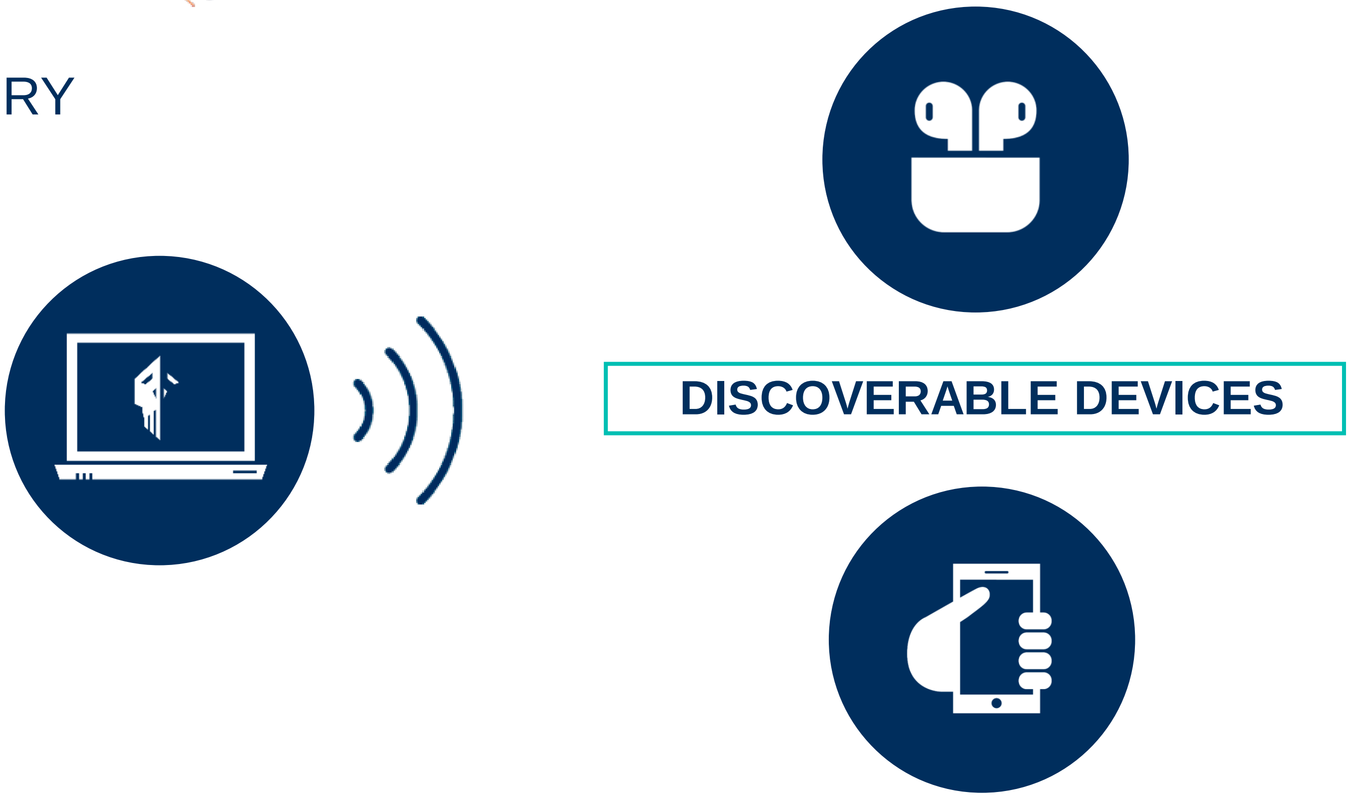
BLUETRUST

#STEPS



BLUETRUST

#DEVICE DISCOVERY



DISCOVERABLE DEVICES

BLUETRUST

#PROFILE CREATION

Name	phone
MAC addres	00:11:22:33:44:55
Device Class	0x0c025a
Version	9
Features	0xfffe8ffed83f5b87
IO Capability	1
Auth. Req.	5

BLUETRUST

#PROFILE CREATION

Some characters are easy to get

Useful tools:

- bluetoothctl
- hciconfig
- wireshark

Name	phone
MAC addres	00:11:22:33:44:55
Device Class	?
Version	?
Features	?
IO Capability	?
Auth. Req.	?

BLUETRUST

#PROFILE CREATION

Others require more work

Useful tools:

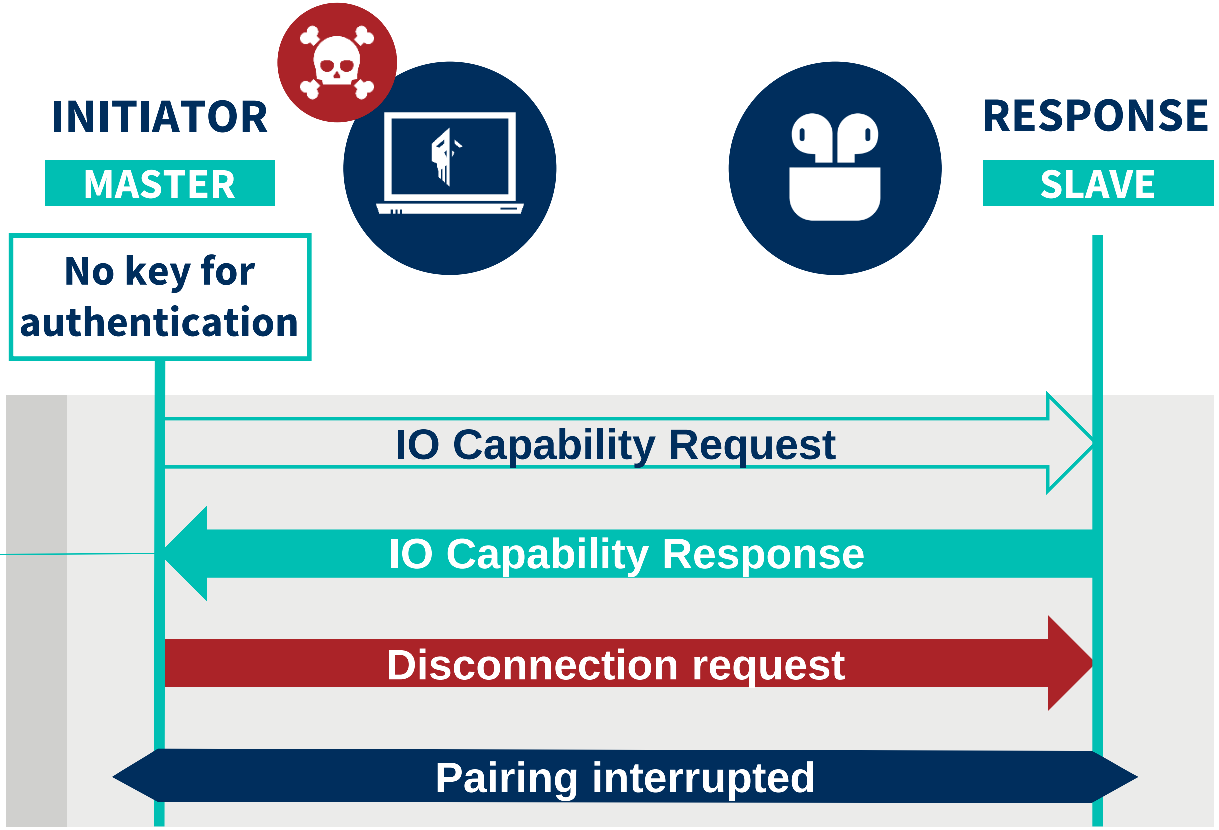
- Scapy

Name	**phone**
MAC addres	**00:11:22:33:44:55**
Device Class	**?**
Version	**?**
Features	**?**
IO Capability	**?**
Auth. Req.	**?**

BLUETRUST

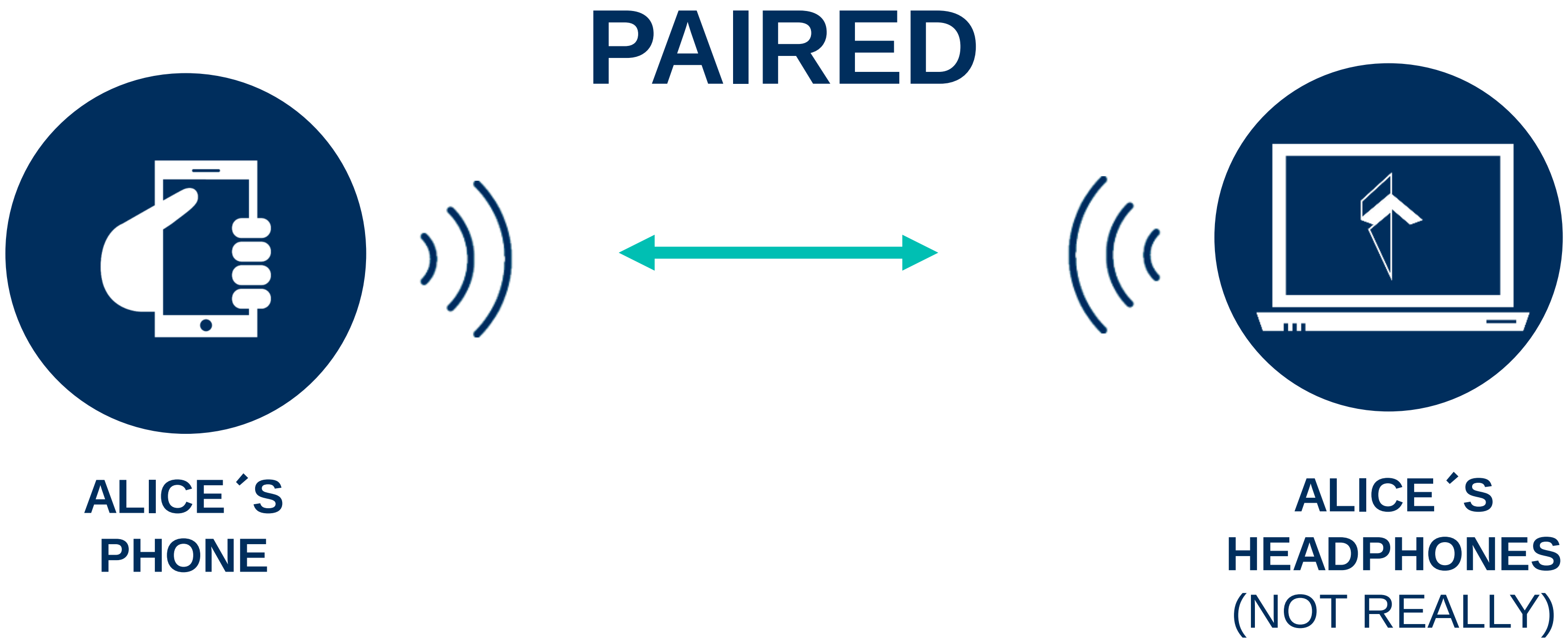
#PROFILE CREATION

IO Capabilities
OOB Data Present
Authentication Requirements



BLUETRUST

#DEVICE IMPERSONATION



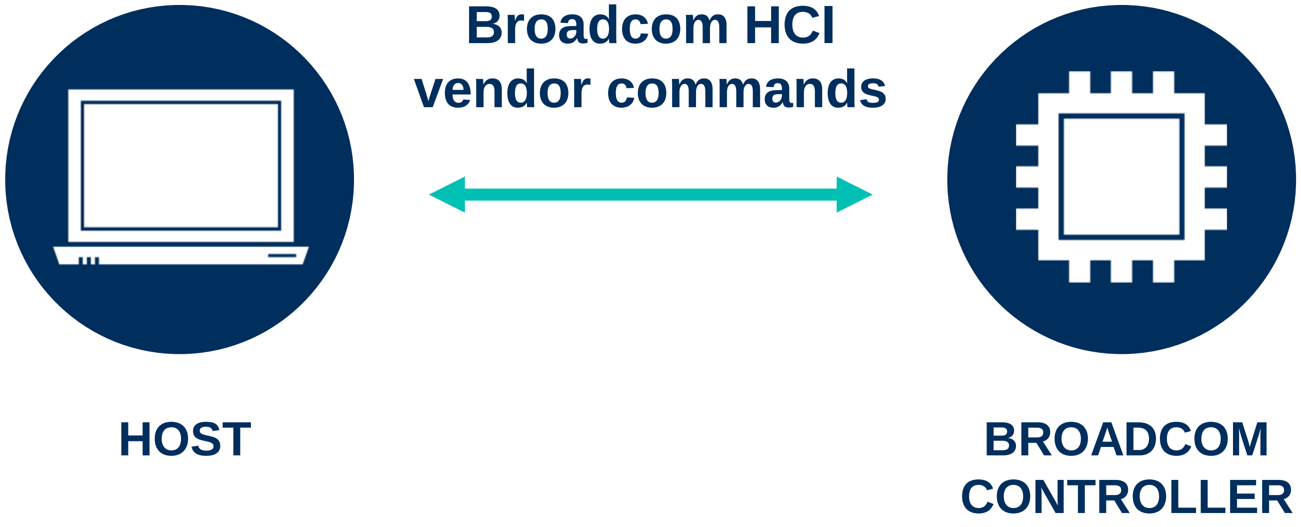
BLUETRUST

# DEVICE IMPERSONATION

BROADCOM VENDOR COMMANDS

- Write data in RAM
- Patch code in ROM

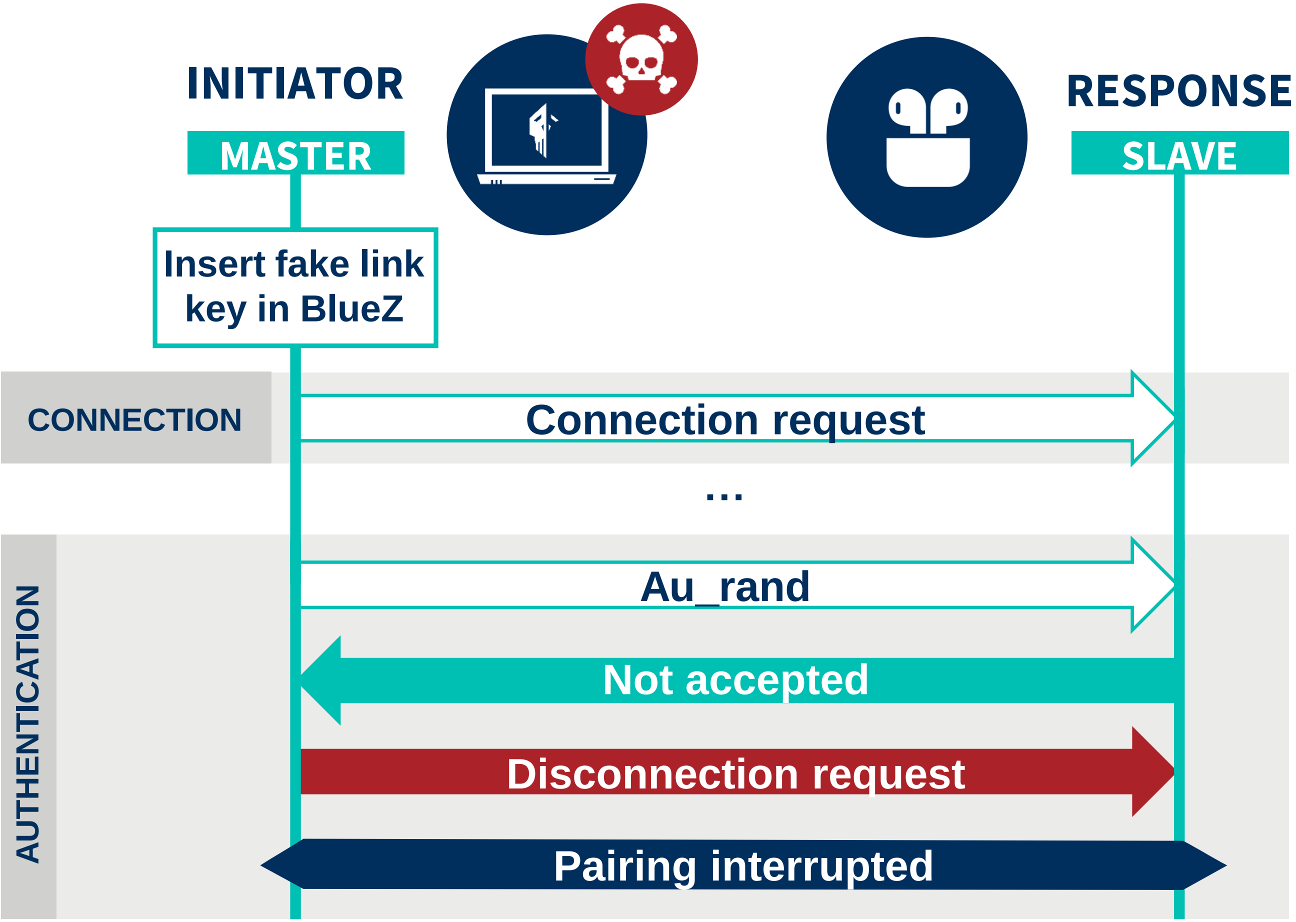
Most is done with **Scapy packets**



BLUETRUST

# PAIRING DETECTION

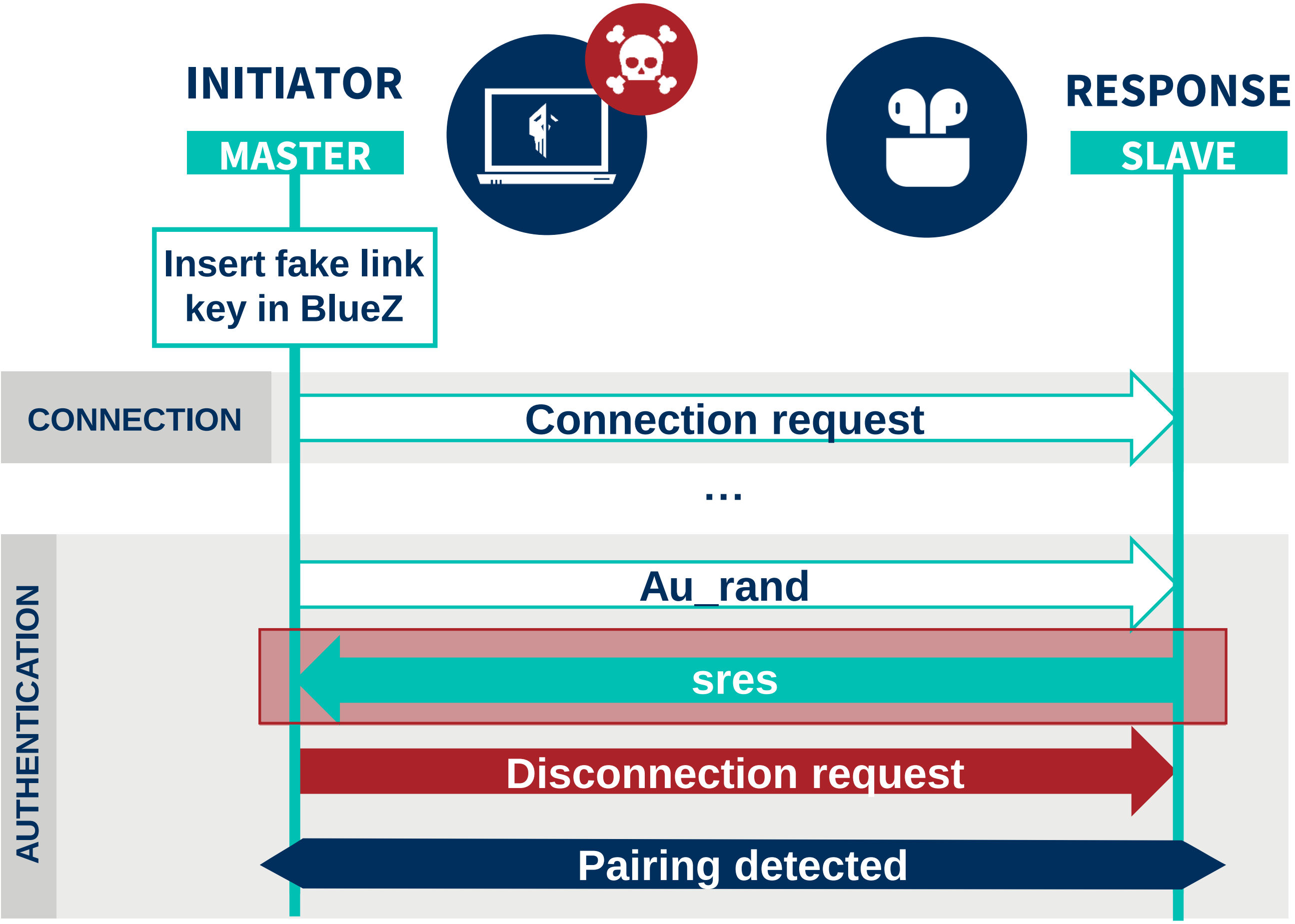
- NEGATIVE DETECTION



BLUETRUST

# PAIRING DETECTION

- POSITIVE DETECTION



BLUETRUST

USES IN CYBERSECURITY

Blue Team

- Perimeter surveillance tool

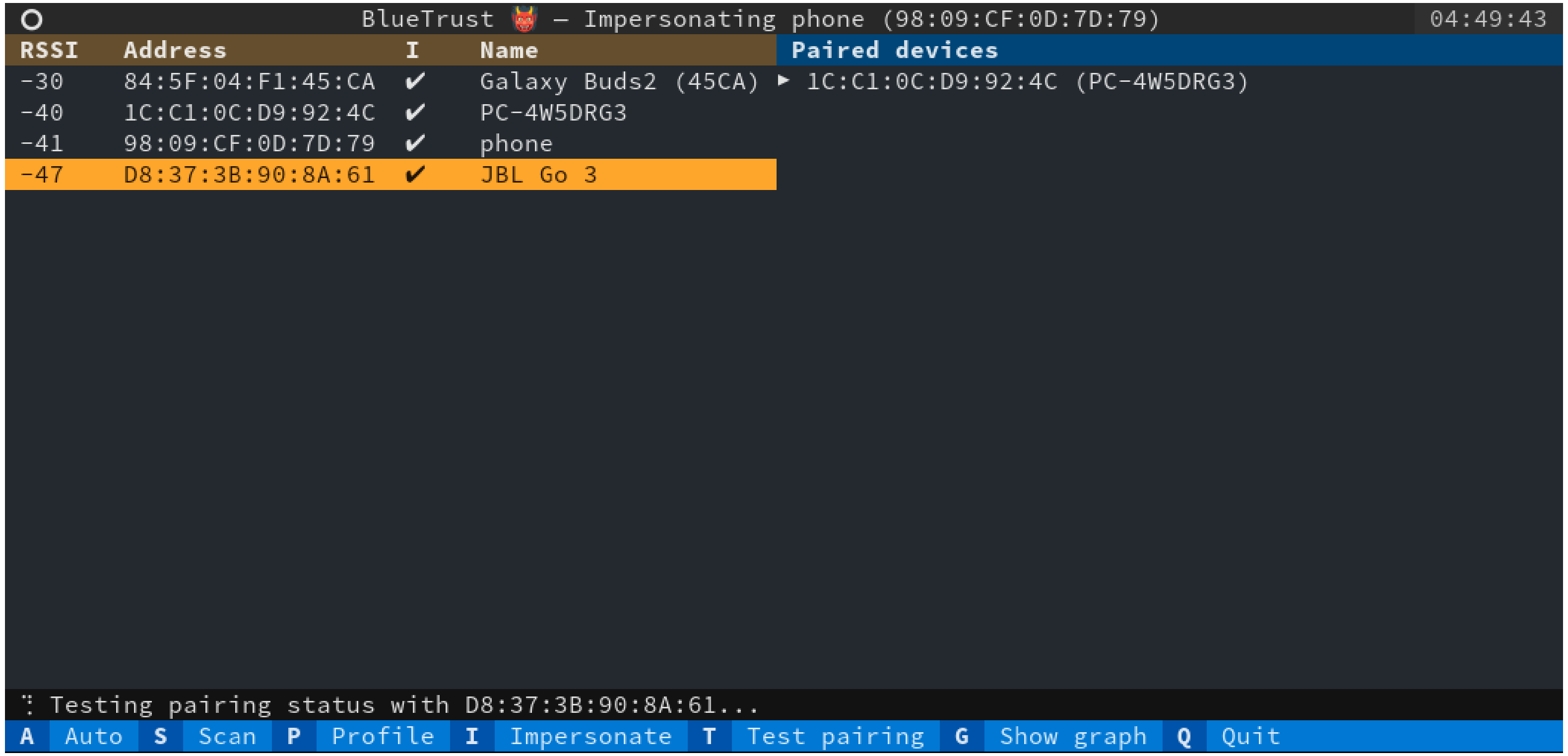
Red Team

- Social engineering information extraction
- Information extraction for physical attacks
- Information extraction to explore the attack surface

Other

- Bluetooth application debugging
- Forensics

THE PoC



```
○                 BlueTrust 👹 — Impersonating phone (98:09:CF:0D:7D:79)               04:49:43
RSSI    Address              I     Name                  Paired devices
-30     84:5F:04:F1:45:CA    ✔     Galaxy Buds2 (45CA)  ▸ 1C:C1:0C:D9:92:4C (PC-4W5DRG3)
-40     1C:C1:0C:D9:92:4C    ✔     PC-4W5DRG3
-41     98:09:CF:0D:7D:79    ✔     phone
-47     D8:37:3B:90:8A:61    ✔     JBL Go 3









⠋ Testing pairing status with D8:37:3B:90:8A:61...
 A  Auto  S  Scan  P  Profile  I  Impersonate  T  Test pairing  G  Show graph  Q  Quit
```

¡MUCHAS GRACIAS!
ESKERRIK ASKO!