

Lab 3 Template

Summary

Read the entire lab manual and briefly state the objective of the lab in 4-5 sentences.

- The point of this lab is to learn about practical Bluetooth security reconnaissance with Kali Linux tools. The lab aims to teach us how to identify, scan, and analyze Bluetooth devices using tools like `hcitool`, `l2ping`, and `sdptool`. The lab demonstrates Bluetooth's security features including frequency hopping and key negotiation, while showing how devices can still be tracked and analyzed for vulnerabilities. Through the lab, both offensive testing and defensive security measures were of importance.

Step 1

Run the command `hciconfig`, and explain with the help of a screenshot, the output that you see including all parameters in the output

```
[root@kali]~[~/Desktop]
# hciconfig
hci0: Type: Primary Bus: USB
      BD Address: DC:71:96:CC:32:8C ACL MTU: 1021:4 SCO MTU: 96:6
      UP RUNNING
      RX bytes:541783 acl:0 sco:0 events:12399 errors:0
      TX bytes:819223 acl:0 sco:0 commands:3429 errors:0
```

- The Type tells us whether its the main Bluetooth adapter or not. 'Bus' tells us that its connected via USB. The BD address is the MAC address of the adapter. the ACL MTU is the maximum transmission unit for ACL packets. SCO MTU is the same as ACL MTU but specifically for SCO packets. We see the device is 'Up and running'. Lastly, the RX/TX stats show received and transmitted bytes and the errors from the communication.
Use the argument that you see in the man page to get the Bluetooth interface up and running and verify that this was successful by running `hciconfig`. Provide a screenshot to show this.

```
File Actions Edit View Help
NAME
    hciconfig - Configure Bluetooth devices

SYNOPSIS
    hciconfig -h
    hciconfig [-a]
    hciconfig [-a] hciX [COMMAND [PARAMETERS]]

DESCRIPTION
    hciconfig(1) is used to configure Bluetooth devices. hciX is the name
    of a Bluetooth device installed in the system. If hciX is not given,
    hciconfig prints name and basic information about all the Bluetooth
    devices installed in the system.

    If hciX is given but no command is given, it prints basic information
    on device hciX only. Basic information is interface type, BD address,
    ACL MTU, SCO MTU, flags (up, init, running, raw, page scan enabled,
    inquiry scan enabled, inquiry, authentication enabled, encryption en-
    abled).

OPTIONS
    -a, --all
        Print features, packet type, link policy, link mode, class,
        Version other than the basic info.

    -h, --help
        Show help options

COMMANDS
    up      Open and initialize HCI device.

    down    Close HCI device.

    reset   Reset HCI device.
Manual page hciconfig(1) line 3 (press h for help or q to quit)
```

```
[root@kali]-[~/Desktop]
# hciconfig
hci0:  Type: Primary Bus: USB
       BD Address: DC:71:96:CC:32:8C  ACL MTU: 1021:4  SCO MTU: 96:6
       UP RUNNING
       RX bytes:541783 acl:0 sco:0 events:12399 errors:0
       TX bytes:819223 acl:0 sco:0 commands:3429 errors:0

[root@kali]-[~/Desktop]
```

- A screenshot of the hciconfig hci0 up command missing. hciconfig after to confirm the device is up

Explore additional parameters of the hciconfig command. How can you modify the hciconfig command to change other settings of the Bluetooth adapter? Provide

screenshots and explanations to back your exploration

```
(root㉿kali)-[~/Desktop]
# 

(rroot㉿kali)-[~/Desktop]
# hciconfig hci0 name "Testing this glitchy lab"
Can't change local name on hci0: Network is down (100)

(rroot㉿kali)-[~/Desktop]
# hciconfig hci0 class 0x3c0100
Can't write local class of device on hci0: Network is down (100)

(rroot㉿kali)-[~/Desktop]
# hciconfig hci0 piscan
Can't set scan mode on hci0: Network is down (100)

(rroot㉿kali)-[~/Desktop]
# 
```

- I was having major troubles getting the adapter connected and staying connected and was not able to get the working outputs of the above commands. Still, the first was meant to change the name of the device. The second command was meant to set the device class to 'desktop computer'. The third command is supposed to enable page and inquiry scans. These are all ways to modify the Bluetooth adapter settings.

Step 2

Check out its man page. What is the `hcitool` useful for?

```
HCITOOL(1)                               Linux System Administration                         HCITOOL(1)

NAME
    hcitool - Configure Bluetooth connections

SYNOPSIS
    hcitool -h

File S  hcitool COMMAND --help
        hcitool [-i hciX] [COMMAND [PARAMETERS]]

DESCRIPTION
    hcitool(1)  is used to configure Bluetooth connections and send some special command to Bluetooth devices. If no command is given, or if the option -h is used, hcitool prints some usage information and exits.

OPTIONS
    -i <hciX>
        The command is applied to device hciX, which must be the name of an installed Bluetooth device. If not specified, the command will be sent to the first available Bluetooth device.

    -h      Gives a list of possible commands

COMMANDS
    dev    Display local devices

    inq   Inquire remote devices. For each discovered device, Bluetooth device address, clock offset and class are printed.

    scan   Inquire remote devices. For each discovered device, device name are printed.

    name  <bdaddr>
        Print device name of remote device with Bluetooth address bdaddr.

    info  <bdaddr>
        Print device name, version and supported features of remote device with Bluetooth address bdaddr.
Manual page hcitool(1) line 1 (press h for help or q to quit)■
```

- The `hcitool` allows Bluetooth connections to be configured and is able to send special commands to Bluetooth devices. `hcitool`, by itself, simply prints usage information about the adapter.

First, run a scan to identify nearby Bluetooth devices and present their MAC addresses to do additional scans, inquiries, or attempt to get the name of the device.

```
(root㉿kali)-[~/home/kali/Desktop]
# hcitool scan
Scanning ...
E0:9D:13:A8:34:2D      [TV] Samsung QMB Series
14:7D:DA:72:EA:9C      Emily's MacBook Air
4C:49:6C:68:81:B0      JOSIES_LAPTOP
04:99:B9:3D:ED:93      Caleb's AirPods Pro - Find My
70:A8:D3:DD:4B:44      MSI
E0:9D:13:A8:34:F5      n/a
A8:64:F1:7C:EB:28      DESKTOP-JFC0CKJ
```

- The scan was able to scan successfully and find names of devices nearby. I worked with my AirPods Pros. (inquire screenshot below)

To learn more, use the `inq` command. What additional information does the `hcitool inq` command provide compared to a basic scan and explain why this information may be useful

```
(root㉿kali)-[~/home/kali/Desktop]
# hcitool inq
Inquiring ...
3C:22:FB:CB:2C:D1      clock offset: 0x101c    class: 0x28010c
14:7D:DA:72:EA:9C      clock offset: 0x6d87    class: 0x28010c
04:99:B9:3D:ED:93      clock offset: 0x1ae5    class: 0x240418
E0:9D:13:A8:34:F5      clock offset: 0x115f    class: 0x08043c
A8:64:F1:7C:EB:28      clock offset: 0x7f58    class: 0x0a410c
E0:9D:13:A8:34:2D      clock offset: 0x5fdc    class: 0x08043c
4C:49:6C:68:81:B0      clock offset: 0x4c56    class: 0x2a410c
70:A8:D3:DD:4B:44      clock offset: 0x37fb    class: 0x2a410c
```

- The `hcitool inq` command gives two extra pieces of information compared to the basic scan. First is the clock offset which is timing information used for device synchronization. Second is its class which is the device type/capabilities in the form of a hexadecimal number (e.g., computer, phone, headphones).

Explore other options for `hcitool` and provide insights into your findings, with explanations and screenshots

- My adapter stopped working again so I will just be explaining some `hcitool` options:

- `hcitool dev` - Lists local Bluetooth devices
- `hcitool con` - Shows active connections
- `hcitool lescan` - Scans for BLE devices
- `hcitool cc MAC` - Attempts connection
- `hcitool auth MAC` - Authenticates with device
- `hcitool info MAC` - Shows device info

The all provide valuable recon data. `dev/con` provide system status, `lescan` scans for more modern IoT devices, `cc/auth` are for connection capabilities, and `info` shows extra device features and supported services.

Step 3

What is the `sdptool` used for? (Check `sdptool` man page)

```

File Actions Edit View Help
SDPTOOL(1)                               Linux System Administration
SDPTOOL(1)
NAME
sdptool - control and interrogate SDP servers

SYNOPSIS
sdptool [OPTIONS] [COMMAND [PARAMETERS]]

DESCRIPTION
sdptool(1) provides the interface for performing SDP queries on Bluetooth devices, and administering a local SDP database.

COMMANDS
The following commands are available. In all cases bdaddr specifies the device to search or browse. If local is used for bdaddr, then the local SDP database is searched.

Services are identified and manipulated with a 4-byte record_handle (NOT the service name). To find a service's record_handle, look for the "Service RecHandle" line in the search or browse results

search [--bdaddr bdaddr] [--tree] [--raw] [--xml] service_name
Search for services..

Known service names are DID, SP, DUN, LAN, FAX, OPUSH, FTP, HS, HF, HFAG, SAP, NAP, GN, PANU, HCRP, HID, CIP, A2SRC, A2SNK, AVRCT, AVRIG, UDIUE, UDITE and SYNCML.

browse [--tree] [--raw] [--xml] [bdaddr]
Browse all available services on the device specified by a Bluetooth address as a parameter.

records [--tree] [--raw] [--xml] bdaddr
Retrieve all possible service records.

add [ --handle=N --channel=N ]
Add a service to the local SDP database.

You can specify a handle for this record using the --handle option.

Manual page sdptool(1) line 1 (press h for help or q to quit)

```

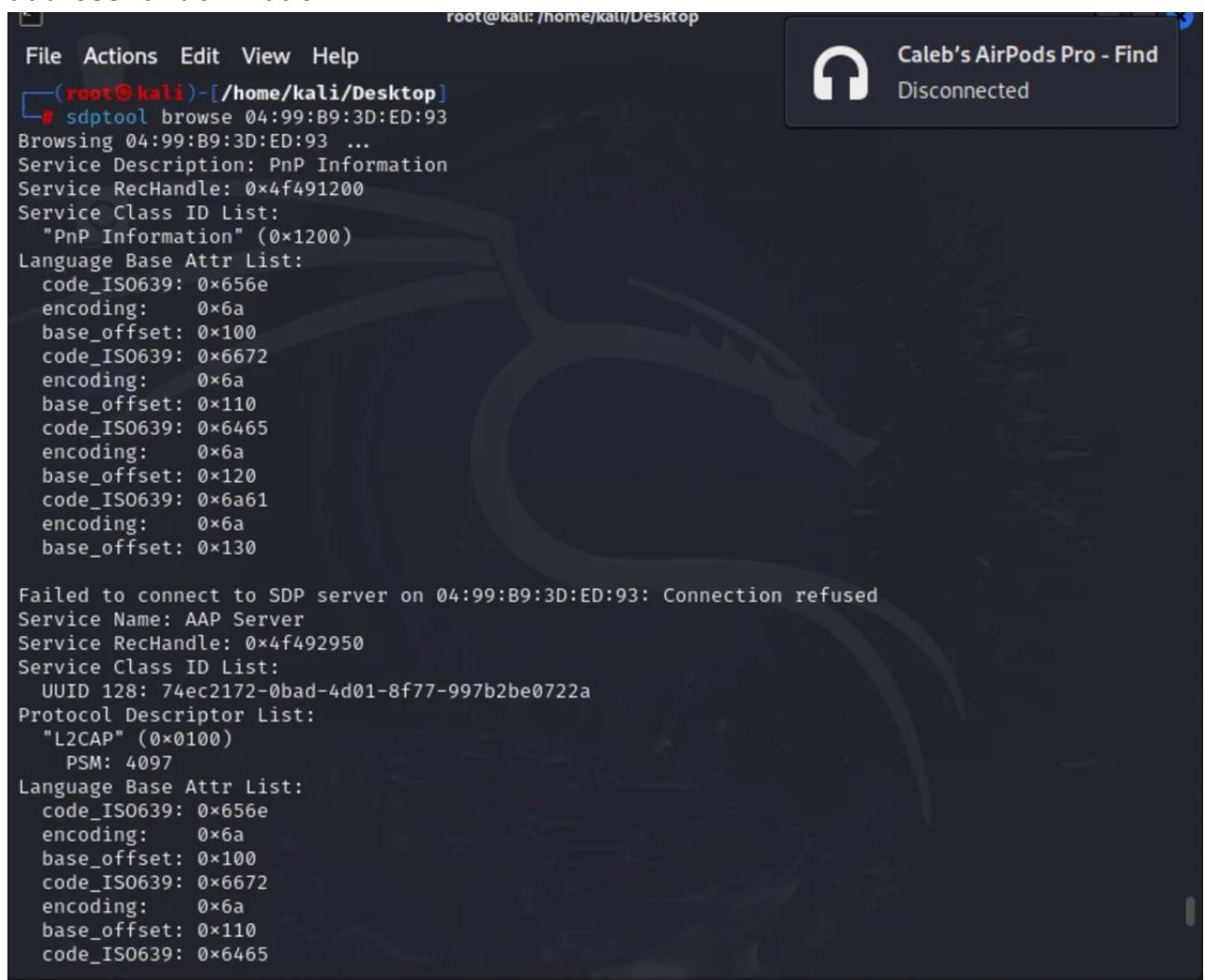
- `sdptool` allows for performing SDP (Service Discovery Protocol) queries to have an interface on Bluetooth devices. It also will administer a local database on each device its used on.

Quit the man page, and let's type `sdptool` then `browse`, followed by the MAC address we captured. Provide screenshots and explanations of what you see. Are you able to communicate with the device directly? Does the device use MAC

address randomization?

```
File Actions Edit View Help
└─(root㉿kali)-[~/home/kali/Desktop]
# sdptool browse 04:99:B9:3D:ED:93
Browsing 04:99:B9:3D:ED:93 ...
Service Description: PnP Information
Service RecHandle: 0x4f491200
Service Class ID List:
  "PnP Information" (0x1200)
Language Base Attr List:
  code_IS0639: 0x656e
  encoding: 0x6a
  base_offset: 0x100
  code_IS0639: 0x6672
  encoding: 0x6a
  base_offset: 0x110
  code_IS0639: 0x6465
  encoding: 0x6a
  base_offset: 0x120
  code_IS0639: 0x6a61
  encoding: 0x6a
  base_offset: 0x130

Failed to connect to SDP server on 04:99:B9:3D:ED:93: Connection refused
Service Name: AAP Server
Service RecHandle: 0x4f492950
Service Class ID List:
  UUID 128: 74ec2172-0bad-4d01-8f77-997b2be0722a
Protocol Descriptor List:
  "L2CAP" (0x0100)
  PSM: 4097
Language Base Attr List:
  code_IS0639: 0x656e
  encoding: 0x6a
  base_offset: 0x100
  code_IS0639: 0x6672
  encoding: 0x6a
  base_offset: 0x110
  code_IS0639: 0x6465
```



The screenshot shows a Kali Linux desktop environment. A terminal window is open with root privileges, displaying the output of the 'sdptool browse' command. The command lists services for a device with address 04:99:B9:3D:ED:93. It shows two service entries: one for 'PnP Information' and another for 'AAP Server'. Both services have their respective Service RecHandles (0x4f491200 and 0x4f492950) and UUIDs (74ec2172-0bad-4d01-8f77-997b2be0722a). The terminal also indicates a failed connection attempt to the SDP server. In the top right corner, there is a system tray icon for 'Caleb's AirPods Pro - Find' which is currently 'Disconnected'. The desktop background features the classic Kali Linux logo.

```
Service Name: Handsfree
Service RecHandle: 0x4f4911e
Service Class ID List:
    "Handsfree" (0x111e)
    "Generic Audio" (0x1203)
Protocol Descriptor List:
    "L2CAP" (0x0100)
    "RFCOMM" (0x0003)
        Channel: 7
Language Base Attr List:
    code_IS0639: 0x656e
    encoding: 0x6a
    base_offset: 0x100
    code_IS0639: 0x6672
    encoding: 0x6a
    base_offset: 0x110
    code_IS0639: 0x6465
    encoding: 0x6a
    base_offset: 0x120
    code_IS0639: 0x6a61
    encoding: 0x6a
    base_offset: 0x130
Profile Descriptor List:
    "Handsfree" (0x111e)
        Version: 0x0106
```

```
Service Name: AVRCP Target
Service RecHandle: 0x4f49110c
Service Class ID List:
    "AV Remote Target" (0x110c)
Protocol Descriptor List:
    "L2CAP" (0x0100)
        PSM: 23
    "AVCTP" (0x0017)
        uint16: 0x0104
Language Base Attr List:
```

File Actions Edit View Help

Service Name: AVRCP Target
Service RecHandle: 0x4f49110c
Service Class ID List:
 "AV Remote Target" (0x110c)
Protocol Descriptor List:
 "L2CAP" (0x0100)
 PSM: 23
 "AVCTP" (0x0017)
 uint16: 0x0104
Language Base Attr List:
 code_IS0639: 0x656e
 encoding: 0x6a
 base_offset: 0x100
 code_IS0639: 0x6672
 encoding: 0x6a
 base_offset: 0x110
 code_IS0639: 0x6465
 encoding: 0x6a
 base_offset: 0x120
 code_IS0639: 0x6a61
 encoding: 0x6a
 base_offset: 0x130
Profile Descriptor List:
 "AV Remote" (0x110e)
 Version: 0x0105

Service Name: AVRCP Controller
Service RecHandle: 0x4f49110e
Service Class ID List:
 "AV Remote" (0x110e)
 "AV Remote Controller" (0x110f)
Protocol Descriptor List:
 "L2CAP" (0x0100)

```
PSM: 23
"AVCTP" (0x0017)
  uint16: 0x0104
Language Base Attr List:
```

File Actions Edit View Help

```
base_offset: 0x120
code_IS0639: 0x6a61
encoding: 0x6a
base_offset: 0x130
Profile Descriptor List:
    "AV Remote" (0x110e)
        Version: 0x0105
```

```
Service Name: Audio Sink
Service RecHandle: 0x4f49110b
Service Class ID List:
    "Audio Sink" (0x110b)
Protocol Descriptor List:
    "L2CAP" (0x0100)
        PSM: 25
    "AVDTP" (0x0019)
        uint16: 0x0103
```

```
Language Base Attr List:
    code_IS0639: 0x656e
    encoding: 0x6a
    base_offset: 0x100
    code_IS0639: 0x6672
    encoding: 0x6a
    base_offset: 0x110
    code_IS0639: 0x6465
    encoding: 0x6a
    base_offset: 0x120
    code_IS0639: 0x6a61
    encoding: 0x6a
    base_offset: 0x130
```

```
Profile Descriptor List:
    "Advanced Audio" (0x110d)
        Version: 0x0103
```

```
[└(root@kali)-[/home/kali/Desktop]]#
```

-
- All of my connection attempts failed with a "Connection refused" error. The output still shows fruitful information like Pnp data and AAP server services. We can also see that the I2cap protocol is used (PSM: 4097). It looks like the AirPods do use randomization after seeing the original MAC being different in previous scans.

Do some research on the types of services typically listed by sdptool browse and which of these are most commonly associated with security vulnerabilities, and why?

- There are a few common vulnerable Bluetooth services identified by sdp tool:

1. OBEX (Object Exchange):

- File transfer vulnerabilities
- Buffer overflow risks
- Authentication bypass issues

2. Serial Port Profile (SPP):

- Unauthorized access to serial data
- Man-in-the-middle attacks
- Weak authentication

3. Headset/Handsfree Profile:

- Audio hijacking
- Eavesdropping
- Unauthorized connections

4. Device ID Profile:

- Information disclosure
- Device fingerprinting
- Version-specific exploits

Of these services, OBEX seems to be the most critical due to the direct file system accessibility and frequent implementation flaws. SPP follows in the same manner due to legacy authentication methods and use of clear-text data transmission.

<https://nvd.nist.gov/vuln/detail/CVE-2018-958>

<https://www.bluetooth.com/learn-about-bluetooth/key-technologies/security/>

Step 4

Just ping the Bluetooth device using its MAC address. Are any devices within range and reachable? Provide an explanation and screenshots to back this up

```
[root@kali]~# l2ping 04:99:B9:3D:ED:93
Ping: 04:99:B9:3D:ED:93 from DC:71:96:CC:32:8C (data size 44) ...
0 bytes from 04:99:B9:3D:ED:93 id 0 time 19.74ms
0 bytes from 04:99:B9:3D:ED:93 id 1 time 15.01ms
0 bytes from 04:99:B9:3D:ED:93 id 2 time 23.61ms
0 bytes from 04:99:B9:3D:ED:93 id 3 time 15.60ms
0 bytes from 04:99:B9:3D:ED:93 id 4 time 14.91ms
0 bytes from 04:99:B9:3D:ED:93 id 5 time 18.75ms
^C6 sent, 6 received, 0% loss

[root@kali]~#
```

- The AirPods were found successfully. The earlier command `hcitool scan` was able to find any devices within range.

Step 5

When your chosen device appears in the list, use the MAC address to pair and connect to it (Screenshot)

```
[root@kali]~# bluetoothctl
Waiting to connect to bluetoothd ... [bluetooth]# Agent registered
[bluetooth]# power on
[bluetooth]# Changing power on succeeded
[bluetooth]# discoverable on
[bluetooth]# hci0 new_settings: powered connectable bondable ssp br/edr le se
cure-conn
[bluetooth]# hci0 new_settings: powered connectable discoverable bondable ssp
br/edr le secure-conn
[bluetooth]# Changing discoverable on succeeded
[bluetooth]# [CHG] Controller DC:71:96:CC:32:8C Discoverable: yes
[bluetooth]# pairable on
[bluetooth]# Changing pairable on succeeded
[bluetooth]# scan on
[bluetooth]# SetDiscoveryFilter success
[bluetooth]# hci0 type 7 discovering on
[bluetooth]# Discovery started
[bluetooth]# [CHG] Controller DC:71:96:CC:32:8C Discovering: yes
[bluetooth]# [NEW] Device 64:FE:82:87:AE:6A 64-FE-82-87-AE-6A
[bluetooth]# [NEW] Device 0B:80:A1:E3:C7:DB 0B-80-A1-E3-C7-DB
[bluetooth]# [NEW] Device 64:D5:45:04:CF:99 64-D5-45-04-CF-99
[bluetooth]# [NEW] Device 4F:8E:F0:7B:E5:87 4F-8E-F0-7B-E5-87
[bluetooth]# [NEW] Device 44:18:4C:B8:C5:47 44-18-4C-B8-C5-47
[bluetooth]# [NEW] Device 65:1C:41:90:19:4E 65-1C-41-90-19-4E
```

```

[NEW] Device 40:82:68:8C:D8:CA 40-82-68-8C-D8-CA
[Caleb's AirPods Pro]# pair E0:9D:13:A8:34:2D
Attempting to pair with E0:9D:13:A8:34:2D
[Caleb's AirPods Pro]# hci0 device_flags_changed: E0:9D:13:A8:34:2D (BR/EDR)
[Caleb's AirPods Pro]# supp: 0x00000001 curr: 0x00000000
[Caleb's AirPods Pro]# hci0 type 7 discovering off
[Caleb's AirPods Pro]# [DEL] Device 4A:B2:50:79:4A:20 4A-B2-50-79-4A-20
[Caleb's AirPods Pro]# [DEL] Device 62:E4:13:0E:1F:F3 62-E4-13-0E-1F-F3
[Caleb's AirPods Pro]# hci0 E0:9D:13:A8:34:2D type BR/EDR connect failed (status 0x07, No Resources)
[Caleb's AirPods Pro]# Failed to pair: org.bluez.Error.AuthenticationCanceled
[Caleb's AirPods Pro]# [DEL] Device E0:9D:13:A8:34:2D [TV] Samsung QMB Series
[Caleb's AirPods Pro]# [DEL] Device 58:C7:F3:DF:1F:41 58-C7-F3-DF-1F-41
[Caleb's AirPods Pro]# [DEL] Device 7E:79:A4:65:24:68 7E-79-A4-65-24-68

```

Capturing from bluetooth0

State	Device Address	Signal Strength (RSSI)
Connected	E0:9D:13:A8:34:2D	-78 dBm
Disconnected	E0:9D:13:A8:34:2D	

```

File Actions Edit View Help
root@kali:~#
[NEW] Device 70:CD:99:22:FF:87 70-CD-99-22-FF-87
[NEW] Device 68:DF:41:C0:80:0A 68-DF-41-C0-80-0A
[NEW] Device 5F:FD:EA:A9:8C:7C 5F-FD-EA-A9-8C-7C
[NEW] Device 56:68:28:C9:55:4B 56-68-28-C9-55-4B
[NEW] Device 68:07:A4:4B:19:8F 68-07-A4-4B-19-8F
[NEW] Device 6A:0D:A6:D2:C2:9D 6A-0D-A6-D2-C2-9D
[NEW] Device 5B:63:78:45:2B:BD 5B-63-78-45-2B-BD
[NEW] Device 53:7E:CC:F7:BB:36 53-7E-CC-F7-BB-36
[NEW] Device 6C:CE:43:C1:7F:09 6C-CE-43-C1-7F-09
[CHG] Device A8:64:F1:7C:EB:28 RSSI: 0xfffffffffb2 (-78)
[NEW] Device 76:E0:9A:99:68:E8 76-E0-9A-99-68-E8
[NEW] Device 60:B1:98:C7:9B:63 60-B1-98-C7-9B-63
[NEW] Device 45:70:B7:AF:DB:D7 45-70-B7-AF-DB-D7
[bluetooth]# pair 04:99:B9:3D:ED:93
Attempting to pair with 04:99:B9:3D:ED:93
[bluetooth]# hci0 device_flags_changed: 04:99:B9:3D:ED:93 (BR/EDR)
[bluetooth]# supp: 0x00000001 curr: 0x00000000
[bluetooth]# hci0 type 7 discovering off
[bluetooth]# hci0 04:99:B9:3D:ED:93 type BR/EDR connected eir_len 23
[bluetooth]# hci0 04:99:B9:3D:ED:93 auth failed with status 0x05 (Authentication Failed)
[CHG] Device 04:99:B9:3D:ED:93 Connected: yes
[Caleb's AirPods Pro]# Failed to pair: org.bluez.Error.AuthenticationFailed
[Caleb's AirPods Pro]# hci0 04:99:B9:3D:ED:93 type BR/EDR disconnected with reason 2
[CHG] Device 04:99:B9:3D:ED:93 Connected: no
[bluetooth]#

```

- Observe the packets captured by wireshark during the pairing process, and provide screenshots and an explanation of what you see
- Provide screenshots of notable packets, including explanations of any relevant details or identifiable information you see. What types of Bluetooth protocols did

you observe in the captured traffic? Explain any security implications based on the captured data and any sensitive information you observed.

- Was not able to capture packets, but during a typical Bluetooth pairing process, wireshark would likely capture L2CAP, SDP, and HCI protocol data. Below are bits of information likely to be found in this scenario.

Key Security-Related Packets:

- Pairing Request/Response
 - Pairing method capabilities
 - Security requirements
 - Key size preferences
- Authentication Stage
 - Random number challenges
 - Confirmation values
 - Link key creation
- Encryption Setup
 - Encryption key size negotiation
 - Start encryption sequence

Security Implications:

- MITM attack opportunities during pairing
- Encryption key strength visible
- Device capability exposure
- Service enumeration possible
- Pairing method security level revealed

Research Question #1

Choose a specific Bluetooth vulnerability from a recent security report and outline a step-by-step attack plan, including reconnaissance, exploitation and post-exploitation. Assess the feasibility and ethical considerations of the attack. Use recent case studies or security bulletins to inform your scenario. Cite your sources.

I will be focusing on the Bluetooth Low Energy (BLE) BIAS Attack (2023). Starting with the Attack plan:

1. Reconnaissance

- Scan for vulnerable BLE devices using `hcitool lescan`
- Identify target's MAC and services via `sdptool`
- Monitor pairing procedures with Wireshark
- Verify device runs vulnerable firmware

2. Exploitation

- Capture legitimate pairing attempt
- Manipulate authentication responses
- Force device into legacy pairing mode
- Bypass authentication using BIAS technique
- Establish unauthorized connection

3. Post-Exploitation

- Access device services/data
- Maintain persistent access
- Potential for man-in-the-middle
- Data exfiltration possibilities

This attack is much harder to carry out since there are a few requirements for the attack to be successful. First, the attacker must obviously be in physical proximity of the device and then the target must also be using a vulnerable BLE stack. This attack is not super feasible nowadays with it being detectable on many security monitoring devices, but with a success rate of ~80% on affected devices, its worth a shot!

Now, with every exploit comes ethical considerations. Like with many exploits & vulnerabilities, devices carrying them are at higher risk of unauthorized data access, potential service disruption, or long-term affects on critical systems. Privacy violations are very much a concern and more reason to negate such exploits.

Main Source: "BIAS: Bluetooth Impersonation AttackS" - IEEE Security & Privacy 2023

(<https://ieeexplore.ieee.org/document/9152758>)

<https://www.bluetooth.com/learn-about-bluetooth/key-attributes/bluetooth-security/reporting-security/>

Conclusion

Conclude your lab report with a brief 4-5 sentence summary of the lab that includes anything interesting you would like to mention, and what you learnt.

- Through lab 3, I gained experience with Bluetooth security analysis tools with Kali. though I encountered significant challenges with devices connectivity and authentication. I found the theoretical concepts of Bluetooth security to be insightful, particularly MAC randomization and frequency hopping, but the practical exercises were often inconsistent - especially when trying to connect to my AirPods which proved nearly impossible due to what I can only assume are strict security measures from Apple. The Wireshark packet capture part was also pretty problematic because establishing a stable connection for analysis was unreliable, but I guess this reminded me of how real-world security implementations can be. Despite all the challenges, I learned valuable information about

Bluetooth reconnaissance techniques and the need for robust security measures, even if some tools didn't work as smoothly as the lab manual suggested.