Use of Out of Band in Bluetooth

Final Technical Document

^Sang Yoon Kim and *^~Vincent Mooney

*Secure Hardware VIP Research Group

**School of Electrical and Computer Engineering, College of Engineering*

^School of Computer Science, College of Computing

~School of Cybersecurity and Privacy, College of Computing

Georgia Institute of Technology

May 3 2023

Table of Contents

# I.	Introduction

Bluetooth has become a popular wireless communication method since its commercial introduction in 1999. As the technology becomes more widely used, it presents a valuable target for hackers as the data being sent across it could be intercepted and compromised. Although some vulnerabilities have been addressed in newer versions of the Bluetooth standard, the Man in the Middle (MitM) attack still remains a concern. Therefore, it is important for individuals and organizations to take appropriate security measures to safeguard their data and prevent unauthorized access.

## I.1	Scenario Overview

This VIP sub-group proposes the following potential scenario. Consider a crowded gym where patrons are allowed to connect their smartphone to one of the public equipment using Bluetooth to receive and store health data (heart rate, calories burned, workout length, etc.). However, Bluetooth connections in public settings are often insecure and could put one's personal information at risk. With most Bluetooth connections pairing through Just Works, which is meant to make the pairing process more accessible, this leaves the patron vulnerable to MitM attacks. With constant mass amounts of pairing processes ongoing, it is easy for an attacker to this attack while also in a hidden manner due to the range of commonly used Bluetooth.

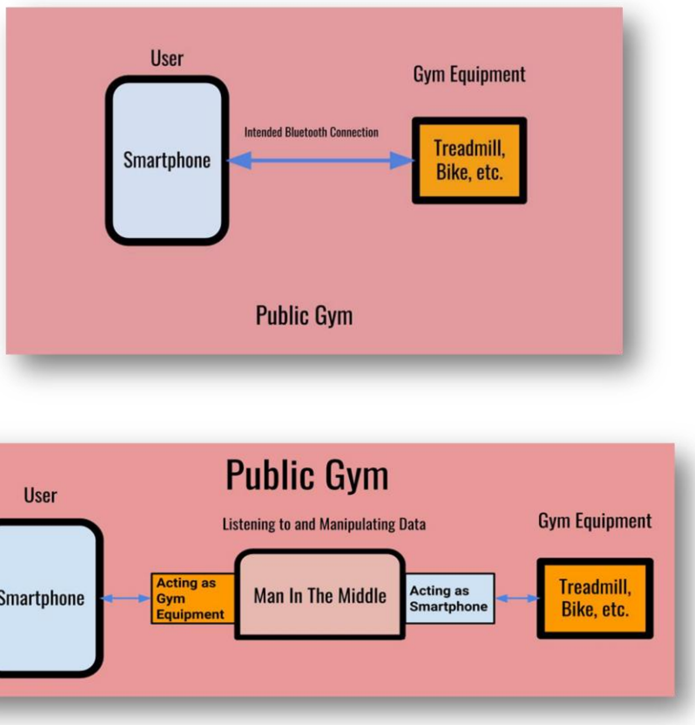


Figure 1. A diagram describing the motivating scenario and how a Man in the Middle could intercept data between two Bluetooth devices.

## I.2	Proposed Solution

Out of Band pairing is a feature described in the Bluetooth standards that allows the pairing process to occur through a non-Bluetooth medium. As can be observed from the scenario, it is crucial to secure the initial pairing process by utilizing an uncompromised medium to do so, other than

Bluetooth. Among the options that exist, Near Field Communication (NFC) is a strong candidate as the short range that the specific communication requires reduces the chance of data interception and, thus, makes it more difficult for a potential attacker to compromise the Bluetooth connection.

This Secure Hardware VIP sub-group aims to research both Bluetooth and NFC's security features and ultimately derive a solution by combining the two methods of communication for a better secure connection between devices. Furthermore, applying NFC as an OOB pairing method will reduce the chance of MitM attacks and prevent low entropy keys during the process. As for the following semester, the sub-group consisted of only a single student who was new to the group, thereby mainly focusing on gaining an understanding of the mechanisms of Bluetooth communication and utilizing the BPA 600 device for confirmation.

II. Terminology

- **MAC Address**: Unique 48-bit identifier for a Bluetooth device [5]
- **BR/EDR**: Basic Rate/Enhanced Data Rate, otherwise known as Bluetooth classic
- **LE**: Low Energy; in addition to Classic Bluetooth, intended to provide reduced power consumption and cost while maintaining a similar communication range [10]
- **ECDH**: Elliptic-Curve Diffie Hellman, a key agreement protocol that allows two parties to establish a shared secret over an insecure channel
- **Link Key**: a secret key known by two devices used to provide confidentiality and authentication to a Bluetooth connection; generated by combining contributions from each device during pairing
- **SSP**: Secure Simple Pairing; added security features to Legacy Pairing to be Federal Information Processing Standards (FIPS) approved
- **MitM**: Man-in-the-Middle Attack, where a 3rd party pretends to be both devices in a Bluetooth connection
- **OOB**: Out-of-Band Pairing, a way to connect Bluetooth devices using a medium other than Bluetooth
- **FIPS**: Federal Information Processing Standards; information technology standards for use within non-military government agencies and by government contractors and vendors working with the agencies
- **STK**: Short Term Keys; temporary keys initially used during the first steps of Legacy Pairing
- **LTK**: Long Term Keys; keys which are stored and used for future connections through Legacy Pairing

III. Bluetooth

III.1 Bluetooth

Bluetooth is a wireless communication technology that operates at the 2.4GHz frequency band. Depending on the range it is capable of communicating, it is divided into three categories – Class 1, Class 2, and Class 3. Due to the trade-off of the range of communication and the necessary power for it, most Bluetooth devices are equipped with Class 2 Bluetooth, which has a range of 10m or 33ft. Within these distances, Bluetooth allows connected electronic devices to send relatively small amounts of data to and from each other. Each device with the capability of connecting through Bluetooth is assigned to a 48-bit address, known as the MAC address. The first significant 24 bits are referred to as Organizationally Unique Identifier (OUI), which is supposedly unique for each manufacturer. The second 24 bits, on the other hand, are intended to be unique to the individual device.

	Range	Power Requirement
Class 1	100 m	100 mW
Class 2	10 m	2.5 mW
Class 3	1 m	1 mW

Table 1. A table describing the range and the power requirement of the three classes that exists for Bluetooth.

III.2 Bluetooth Special Interest Group (SIG)

The Bluetooth SIG is a standards organization founded in 1998, which covers the development of Bluetooth standards and licensing of Bluetooth technologies and trademarks to manufacturers. The board of directors, which is the group having potential power over the whole organization, consists of one representative from each Promoter member company - Microsoft, Intel, Toshiba, Telink Semiconductor, Nokia, Google, Ericsson AB, Motorola, Apple - plus up to four Associate Member Directors. The Promotor member companies additionally have considerable influence over the strategic and technological directions of Bluetooth.

The internal structure of Bluetooth Sig consists of Study groups, Expert groups, Working groups, and Committees. While each group has its individual roles, which leverage the actions of Bluetooth SIG in general, the Working group may be considered the most significant. As the Working groups develop new Bluetooth specifications and enhance adopted specifications, they are responsible for most published standards and specifications. It is worth noting that participation in the Working group is restricted to only Promoter members and Associate member companies.

III.3 Current Standards

As this research was being held, the most recent Bluetooth version is Bluetooth 5.3, released in July 2021 by Bluetooth SIG. The specific update focused on four significant modifications: enhancement in data transmission during periodic advertising, enhancement over encryption key size negotiations, allowing specific channel selection when adaptive frequency hopping, and enhancement over sudden change between power saving mode and higher bandwidth usage. However, as there were no major updates on the security of Just Works or any other pairing methods, it is safe to assume that the danger of MitM attacks still persists.

While there has not been any official release from Bluetooth SIG, it is said that Bluetooth 5.4 will be released sometime soon, in 2023. According to Bluetooth SIG, the update is expected to include changes in advertising coding selection, encrypting advertising data, and periodic advertising with responses. Similar to the above, as there has been no particular announcement or research published on the security issue regarding pairing, it is safe to assume the vulnerability of the MitM attack will remain. It is also important to note that this will most likely be a consistent problem, as even if Bluetooth SIG addressed the issue and derived a potential solution, it would result in more harm than good for the companies participating in the organization due to the massive amount of faulty Bluetooth equipped devices losing their ability of backward compatibility.

III.4 Pairing Process

While there are multiple different methods of pairing two Bluetooth devices, each method still follows a general multi-step pairing process to share information. The process begins with a searching device, which sends out an inquiry to find nearby discoverable devices. A found device will respond to the inquiry with its Bluetooth Address, initiating the pairing stage of the process. During pairing, the searching device and the found device will send each other their Bluetooth Addresses. Pairing ends, and a connection is established by creating a 128-bit pairing key to authenticate the link once the Bluetooth Addresses are confirmed. Connected devices can either actively send data to each other, or go into a sleep mode – only equipped with Bluetooth LE -where data transmission is inactive. Sleep mode ends when devices are prompted to send data again. Unless the initial pairing packets were set not to do so, connected devices will store each other's information in memory and automatically connect when discoverable in range.

III.5 Pairing Methods

There are three methods of pairing that Bluetooth provides when two devices initially attempt to connect: Legacy pairing, Secure Simple pairing, and Secure Connection. Legacy pairing is the method that was provided originally by Bluetooth SIG when Bluetooth was first introduced. It uses a simple process of exchanging data to derive a symmetric key with which to encrypt the data transmission during the STK and LTK distribution phase. Although, at the time of release, the aim was to simply provide any sort of a secure method to assist the pairing process, it was not FIPS approved. The method allows Just Works, Passkey Entry, and Out of Band to be implemented and used if needed.

Secure Simple pairing is a pairing method later released by Bluetooth SIG in Bluetooth 2.1. After having Bluetooth communication settle down to some extent in the market, they aimed to come up with a more secure method of pairing that could also be accepted by officials. Thus, by making modifications to Legacy pairing, such as increasing the pin digits from four to six, they were able to achieve the requirements of FIPS. The method allows Just Works, Passkey Entry, Out of Band, and additionally Numeric Comparison (not the case for LE Secure Simple pairing) to be implemented and used if needed.

As Bluetooth SIG started to acknowledge some parts of the fault in the pairing process of Bluetooth, they have released Secure Connection in Bluetooth 4.1 (to LE in Bluetooth 4.2). To create a second line of defense during the pairing process, Secure Connection uses elliptic curve public key cryptography to allow the initial symmetric key to be derived. That key is then used to derive the LTK to be used during Bluetooth data transmission. Similar to Secure Simple pairing, the method allows Just Works, Numeric Comparison, Passkey Entry, and Out of Band to be implemented and used if needed.

Since all three methods, which are still commercially used, share the similarity of transmitting and receiving data for creating the link key (STK and LTK) publicly, they fundamentally possess the vulnerability to a MitM attack. However, since LTK is later used for the current and any future Bluetooth communications between the two devices pairing, securing this process of creating the link key is crucial.

III.6 Pairing Request and Response Packet

Field	Code (1 Byte)	IO Cap. (1 Byte)	OOB Data Flag (1 Byte)	AuthReq (1 Btye)	Max Encryption Key Size (1 Byte)	Initial Key Distribution (1 Byte)	Responder Key Distribution (1 Byte)

Figure 2. A chart describing the structure of a Pairing request and response packet.

The initial packets that are sent between the two devices trying to connect via Bluetooth are referred to as Pairing Request packets and Pairing Response Packets. Both are a 7-byte size packet consisting of information on how the pairing process will take place and which particular circumstances are necessary. As can be seen from the figure, they are divided into seven parts, each representing specific information, sizing up to 1 byte. Code indicates either pairing request or pairing response; IO Cap., otherwise I/O Capabilities, indicates which abilities the device has on communication; OOB Data Flag indicates external means of communication; AuthReq, otherwise Authentication Request, indicates the want for options such as MitM protection, generation of long-term keys, etc.; Max Encryption Key Size indicates the maximum key size to be used; Initial & Responder Key Distribution indicates the type of keys a device may provide or request, such as an LTK.

III.7 IO Cap. And Pairing Methods

One of the parts of the Pairing Request and Response Packet is IO Cap. This part of the packet is worth noting for this research as it shows the device's capabilities for authentication during the pairing process. Mainly there are two factors - input and output – considered when assigning the value for this part. Input-wise, specifications such as the device having a keyboard, a simple yes/no button, or having no ability to input at all are considered. Output-wise, specifications on whether the device has a display or not are considered. As can be seen from the chart, depending on the input and output, each device is assigned a particular value representing its state. When pairing, the IO Cap. values from both devices are compared, and the decision of which specific method of authentication - Just Works, Passkey Entry, Numeric Comparison, or OOB – the following pairing will use.

Value	Description
0x00	DisplayOnly
0x01	DisplayYesNo
0x02	KeyboardOnly
0x03	NoInputNoOuput
0x04	KeyboardDisplay
0x05-0xFF	Reserved

Table 2. A table describing the potential values of IO Cap. and its description of the capabilities.

IV. Out of Band (OOB) Pairing

OOB pairing is a feature described in the Bluetooth Standard that allows the pairing process to occur over a non-Bluetooth medium. The discovery information, such as Bluetooth Addresses, is exchanged through this medium instead of sharing information in the open. While the specific medium that could be used for OOB is up to the developer, all methods have to create a temporary key after authentication. Creating this key ends the stage of OOB and is then used for subsequent steps in creating the STK and LTK. Both devices in the pairing process must set their OOB flag to initiate OOB pairing. If the correct values are not set in the Pairing Request, and Response packets, OOB pairing will not be initiated.

Value	Description
0x00	OOB Authentication data not present
0x01	OOB Authentication data from remote device present
0x02-0xFF	Reserved

Table 3. A table describing the potential values of OOB and its meaning during packet transfers.

V. Just Works

Just Works is one of the authentication methods that is provided by Bluetooth, when comparing the IO Cap. indicated that neither IO Cap indicates Passkey entry, Numeric comparison nor OOB could be used. It is assumed as a final option since entropy is non existent in its process. As said by Bluetooth SIG themselves, "Just Works pairing offers no authentication and, hence, no protection against MitM attacks." While other authentication methods have also previously have results of being broke through by methods such as bruteforce, introducing Secure Connection had made it difficult as the method had required the process to take rigorous checking procedures. While this partially solved problems for Passkey Entry and Numeric Comparison, that has not been the case for Just Works as it takes in no form of user input during its process. Thus, it remains as the single most vulnerable authentication method during Bluetooth pairing, and is essential to solve the vulnerability of MitM attack. In order to do so, background research had been done and the following section explains how Just Works provides a flaw to all pairing methods Legacy pairing (Secure Simple pairing) and Secure Connection.

V.1 MitM vulnerability with Just Works - Legacy Pairing

While connecting through Legacy pairing using Just Works, the process can be divided into two main steps: authentication and STK creation. During the authentication step, the two devices, the central and peripheral device, will create a Temporary Key (TK) of 128 bits, as if any other authentication method does so. However, because the method is limited to Just Works, the value of the temporary key will always result to 0. After that, both devices will create and exchange random values denoted as $Rand_c$, $Rand_p$, and use them to derive C_c, C_p through the following function,

C_c = c(TK, $Rand_c$, Pairing Request command, Pairing Response command, address $type_c$, $address_c$, address $type_p$, $address_p$)

After calculating each value, C_c and C_p are then exchanged to check authenticity.

After authenticating the central and peripheral devices in the Bluetooth connection, the STK is created. By using the previous values TK, $Rand_c$, and $Rand_p$, and inserting them into the following function,

STK = s(TK, Randc, Randp)

the STK can be calculated, which will then be used to create the link key for the initial Bluetooth communication session. Despite it being optional, most Bluetooth connections will also have a follow up step of creating a LTK which will be used to create link keys in future sessions between the same central and peripheral device.

V.2 MitM vulnerability with Just Works – Secure Connection

As mentioned in the previous sections, Secure Connection is considered more secure than other pairing methods as it adds in an additional encryption through the Diffie Hellman process. In this case, the pairing process can be divided into three main steps: Diffie Hellman Key (DHKey) creation, first authentication, and second authentication. Initially, following the general structure of the Diffie Hellman procedure, the central and peripheral devices each creates their own secure keys, denoted as $SKey_c$ and $SKey_p$, and also create public keys, denoted as $PKey_c$, $PKey_p$, which are then exchanged between each other. With the necessary information equipped, the values are inserted into the following function to calculate the DHKey.

DHKey = P256(SK_c, PK_p)

While this procedure may prevent passive MitM attacks, it still shows to be vulnerable when the attacker takes additional minimal efforts. As can be seen from the figure, an attacker could potentially create $SKey_1$, $SKey_2$, $PKey_1$, $PKey_2$ and ends up creating two separate keys $DHKey_1$, $DHKey_2$ between the central and peripheral devices individually. After doing so, one will still have the same result of obtaining the DHKeys which will then be used in the latter steps of authenticating the two devices. While this fault in security is not from the structure of Bluetooth pairing itself but from the weakness of the Diffie Hellman procedure in general, it still results in a failure of securing the pairing process of Bluetooth connection.

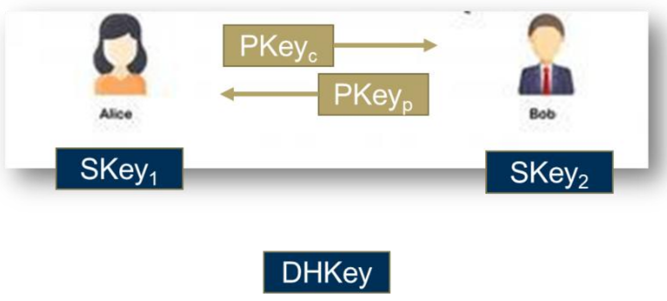


Figure 3. A diagram describing a normal Diffie Hellman key exchange procedure.

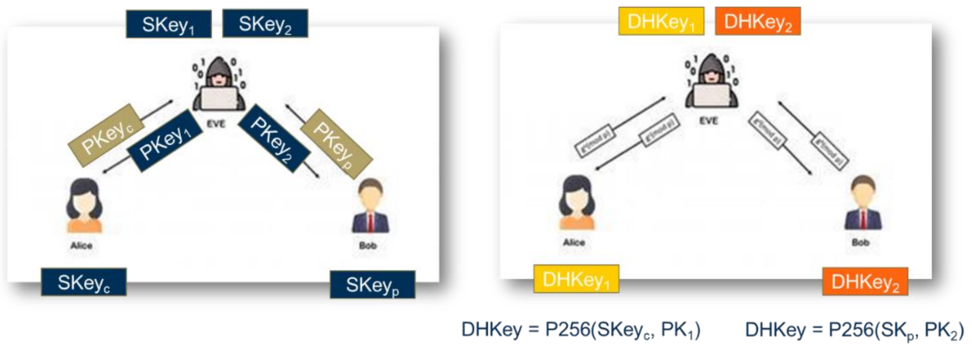


Figure 4. A diagram describing an attacker successfully infiltrating into a Diffie Hellman key exchange procedure.

Typically, after creation of the DHkey, the central and peripheral device will each create and exchange a pseudo-random 128-bit number specifically referred as nonce, denoted as N_c, N_p. Then, the devices will use the nonce values as well as the public keys made from the previous steps to derive C_c, C_p through the following function,

$$C_c = f1(PK_c, PK_p, N_c, 0)$$

After calculating each value, C_c and C_p are then exchanged to check authenticity.

Following the first authentication step, a second authentication also occurs where the LTK is then created. With the previous values of DHKey, N_c, N_p, and Bluetooth addresses obtained, the LTK as well as a MacKey is created through inserting the values in the following function.

$$MacKey \,||\, LTK = f2(DHKey, N_c, N_p, address_c, address_p)$$

The MacKey is then used to create an AES-CMAC value, denoted as E_c, E_p, which are then compared for another authentication. Only after the previous step pass, can the LTK be used for Bluetooth connections.

VI. BPA 600

In order to confirm and check the understandings on Bluetooth done by background research, it was preferable to use a Bluetooth protocol Analyzer to dissect and examine a certain Bluetooth communication. The Bluetooth protocol analyzer, otherwise known as a Bluetooth sniffer, is a chipset hardware device that intercepts, captures, and analyzes Bluetooth signals and their data by occupying the 2.4Ghz ISM band. The general purpose of this hardware device is to capture, decode, and analyze Bluetooth packets in real time for the purpose of debugging or snooping. As the following matched the descriptions of what was necessary in order to fully dissect a Bluetooth connection, the subgroup had proceeded to design an experiment which would use a Bluetooth sniffer.

The specific hardware used as the Bluetooth sniffer was the Teledyne Lecroy Comprobe BPA 600. It is operable on devices that are up to and including Bluetooth 4.2 specification, and is USB powered device that uses two detachable antennas to sniff both LE and Classic Bluetooth connections. The hardware provides 4 different sniff modes which are LE, Classic, Dual Mode (LE and Classic), and Classic-only Multiple Connections. The device allows the user to input the keys (LTKs) directly or in some cases, the software can generate the keys on its own. While in Classic connections sniff mode, the BPA 600 can decrypt Bluetooth packets by receiving inputs on the PIN code involved in the Passkey Entry method during the authentication step, or by receiving the Link Key directly. In LE connections sniff mode, the BPA 600 can receive the LTK directly or the values from OOB method if used, in order to decrypt the packets.

Another aspect of the BPA 600 hardware that is worth noting is the Audio Expert System. . The following system equipped on the hardware allows it to detect and report audio and its potential impairments from any Bluetooth packets. Thereby, users can intercept complete audio transmissions communicated over Bluetooth and can be used to eavesdrop over a Bluetooth connections.

Picture 1. A photo of a BPA 600 hardware.

VII. Experiments

The experiments focused on testing the abilities of the BPA 600 and thus, confirm the sub-group's understandings on the connection of Bluetooth. Therefore, enabling a firm Bluetooth connection was necessary in order to proceed with the rest of the steps of recording and analyzing any Bluetooth packets. However, because the majority of the Bluetooth connections initially planned or done by the previous VIP team were either failing or outdated, multiple experiments were made with different attempts of Bluetooth connection for the BPA 600 to finally analyze.

VII.1 Experiment 1:

The initial experiment that was planned was a Bluetooth connection between a Raspberry Pi and a Raspberry Pi. The reason was because of the freedom(자유도) of being able to choose data to transfer through connection and the relatively easy accessibility of the critical information on Bluetooth connection such as LTKs and pairing packets. With that said, the following experiement aimed to connect two Raspberry Pis equipped with a Bluetooth dongle supporting Bluetooth 4.0. Although there exists a Bluetooth UI for the Rasp OS to easily connect through Bluetooth, the experiment connected by using "bluetoothctl," as the particular method allows the user to alter specific conditions in the pairing process such as manually controlling the IO Cap for the initial advertising packet. The following steps were taken.

- **Enabling certain IO Cap**

1. Open a terminal. Run "bluetoothctl agent off"
2. Run "bluetooth agent NoInputNoOutput"
3. Run "bluetooth agent on"

- **Pairing devices**

1. After agent is on, run "bluetoothctl pairable on"
2. Run "hcitool scan" and look for the MAC address of the other Raspberry Pi device
3. Run "bluetoothctl pair [MAC address]"

Theoretically, "turning on the agent" would allow the pairing process to begin as the device will start to advertise through its Bluetooth packet. However, despite giving the correct message that agent was on, the IO Cap were never updated properly as intended, and connection between both Raspberry Pi devices were never successfully made.

VII.2 Experiment 2:

With methods from experiement 1 failing, an alternative for the recipient of the Raspberry Pi was necessary. Thus, following second attempt aimed for a connection between a Raspberry Pi and an Android device. The particular Android device used for the experiment was equipped with an Android 12 OS, chosen because there exists documents by the previous VIP group that were able to make a Bluetooth connection and retrieve the necessary information to analyze through the BPA 600. Based on those documents, the experiment 2 focused on creating a firm Bluetooth connection, and send certain files via Bluetooth for the BPA 600 to capture and decrypt. The following steps were taken.

- **Pairing devices**

1. Either through steps mentioned in Experiment 1 or through Rasp OS UI, enable pairing and search for the Android device
2. Enable pairing for the Android device and search for Raspberry Pi
3. Connect both devices (without changing any IO Cap, should pair via Numeric Comparison)

While the Bluetooth connection itself was successfully made, the latter part of the intended experiment did not go successfully. Despite multiple attempts to find methods and install certain software to transfer specific files or data over Bluetooth, most of them turned out to be either outdated or not working. Thus, there was no option for the BPA 600 to anaylze any communication between the two devices.

VII.3 Experiment 3:

While experiment 2 was successful on attaining a firm Bluetooth connection, it lacked on obtaining a method to communicate data over Bluetooth for the BPA 600 to analyze. In order to solve this problem and also to utilize one of the main functions of the BPA 600, the Audio Expert System, experiment 3 was taken, focusing on a connection between a Raspberry Pi and a sound device. The particular sound device used in the experiement was a Samsung Galaxy Buds 2 Pro, but theoretically any sound device would work as well. This is because the Raspberry Pi uses the external Bluetooth dongle which supports only up to Bluetooth 4.0, therefore limiting any Bluetooth connection between a sound device to Secure Simple Pairing through Just Works. The aim of the experiment was to initially enable a Bluetooth connection between the two devices, and send specifically sound data for the BPA 600 to capture and decrypt. The following steps were taken.

- **Pairing devices**

1. Either through steps mentioned in Experiment 1 or through Rasp OS UI, enable pairing and search for the Sound device
2. Enable pairing for the Sound device and search for Raspberry Pi
3. Connect both devices (without changing any IO Cap, should pair via Numeric Comparison)

- **Deriving the Link Key**

1. Open a terminal. Run "sudo su"
2. Once pairing is made through the above step, navigate to "/var/lib/Bluetooth/[Raspberry Pi's MAC address]/[Sound device's MAC address]"
3. Open the info file "cat info " and find the link key

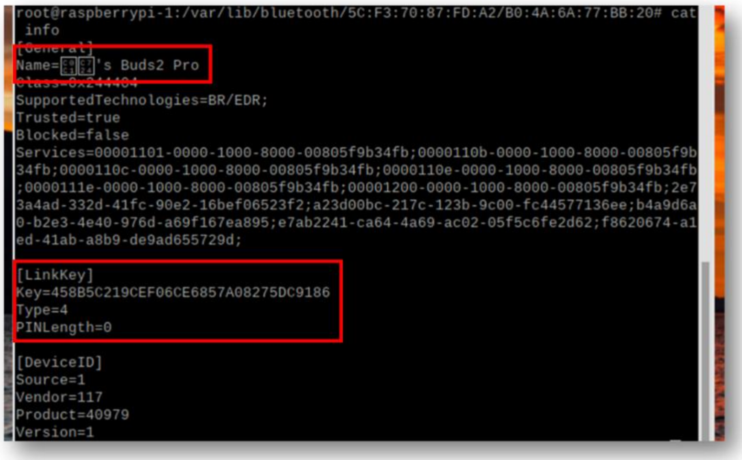
- **Preparing and capturing communication**

1. Through BPA 600 software, scan the necessary Bluetooth devices
2. Set the capture method to Bluetooth Classic, and select the right devices
3. Select Link Key as the Classic Encryption and insert the correct Link key in the Value
4. Record and start connection, play audio
5. End record and save the capture

- **Decrypting the packets**

1. Through BPA 600 software, analyze the captured files through frame display

2. For audio transmission, use Audio Expert System to decrypt the data straight forward

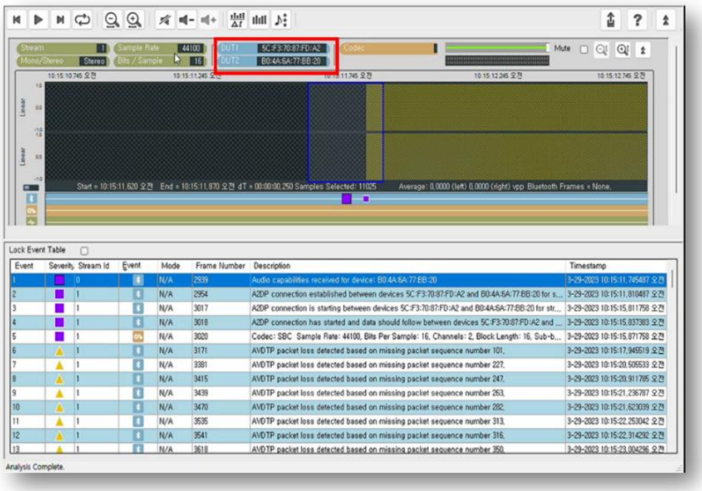


```
root@raspberrypi-1:/var/lib/bluetooth/5C:F3:70:87:FD:A2/B0:4A:6A:77:BB:20# cat
 info
[General]
Name=□□'s Buds2 Pro
Class=0x244404
SupportedTechnologies=BR/EDR;
Trusted=true
Blocked=false
Services=00001101-0000-1000-8000-00805f9b34fb;0000110b-0000-1000-8000-00805f9b
34fb;0000110c-0000-1000-8000-00805f9b34fb;0000110e-0000-1000-8000-00805f9b34fb
;0000111e-0000-1000-8000-00805f9b34fb;00001200-0000-1000-8000-00805f9b34fb;2e7
3a4ad-332d-41fc-90e2-16bef06523f2;a23d00bc-217c-123b-9c00-fc44577136ee;b4a9d6a
0-b2e3-4e40-976d-a69f167ea895;e7ab2241-ca64-4a69-ac02-05f5c6fe2d62;f8620674-a1
ed-41ab-a8b9-de9ad655729d;

[LinkKey]
Key=458B5C219CEF06CE6857A08275DC9186
Type=4
PINLength=0

[DeviceID]
Source=1
Vendor=117
Product=40979
Version=1
```

Picture 2. A screenshot of a Raspberry Pi showing the Bluetooth connection data between a sound device.

VIII. Results

Experiment 3 ran successfully, and various parts of the knowledge on Bluetooth communication could be checked and confirmed. Firstly, the BPA 600 correctly spotted and recorded the Bluetooth communication between the two intended devices, as can be seen by checking the MAC addresses of the Bluetooth communication packets. More importantly, however, the Audio Expert System was able to decrypt the audio communication packets in real time and create an audio file of the encrypted sound data sent from the Raspberry Pi to the sound device.



```
Start = 10:15:11.620 오전   End = 10:15:11.870 오전 dT = 00:00:00.250 Samples Selected: 11025      Average: 0.0000 (left) 0.0000 (right) vpp  Bluetooth Frames = None.
```

Lock Event Table

Event	Severity	Stream Id	Event	Mode	Frame Number	Description	Timestamp
1	■	0	🟦	N/A	2939	Audio capabilities received for device: B0:4A:6A:77:BB:20	3-29-2023 10:15:11.745487 오전
2	■	1	🟦	N/A	2954	A2DP connection established between devices 5C:F3:70:87:FD:A2 and B0:4A:6A:77:BB:20 for s...	3-29-2023 10:15:11.810487 오전
3	■	1	🟦	N/A	3017	A2DP connection is starting between devices 5C:F3:70:87:FD:A2 and B0:4A:6A:77:BB:20 for str...	3-29-2023 10:15:15.811758 오전
4	■	1	🟦	N/A	3018	A2DP connection has started and data should follow between devices 5C:F3:70:87:FD:A2 and ...	3-29-2023 10:15:15.837383 오전
5	■	1	🟧	N/A	3020	Codec: SBC Sample Rate: 44100, Bits Per Sample: 16, Channels: 2, Block Length: 16, Sub-b...	3-29-2023 10:15:15.871758 오전
6	▲	1	🟦	N/A	3171	AVDTP packet loss detected based on missing packet sequence number 101.	3-29-2023 10:15:17.945519 오전
7	▲	1	🟦	N/A	3381	AVDTP packet loss detected based on missing packet sequence number 227.	3-29-2023 10:15:20.505533 오전
8	▲	1	🟦	N/A	3415	AVDTP packet loss detected based on missing packet sequence number 247.	3-29-2023 10:15:20.911785 오전
9	▲	1	🟦	N/A	3439	AVDTP packet loss detected based on missing packet sequence number 263.	3-29-2023 10:15:21.236787 오전
10	▲	1	🟦	N/A	3470	AVDTP packet loss detected based on missing packet sequence number 282.	3-29-2023 10:15:21.623039 오전
11	▲	1	🟦	N/A	3535	AVDTP packet loss detected based on missing packet sequence number 313.	3-29-2023 10:15:22.253042 오전
12	▲	1	🟦	N/A	3541	AVDTP packet loss detected based on missing packet sequence number 316.	3-29-2023 10:15:22.314292 오전
13	▲	1	🟦	N/A	3618	AVDTP packet loss detected based on missing packet sequence number 350.	3-29-2023 10:15:23.004296 오전

Analysis Complete.

Picture 3. A screenshot of the Audio Expert System showing the result of the decrypted audio data transmitted between the Raspberry Pi and the sound device.

IX. Future Work

Throughout this semester, the sub-group has focused on understanding the mechanisms of Bluetooth communication and how it operates behind the scenes. For that reason, the result is concentrated on confirming the group's personal understanding of the technology and assessing how much of the previous equipment can be reused to the current date. While the previous VIP team in 2020 who worked on the BPA 600 paid great attention to the device's ability to decrypt connection without providing information to the encryption process, I highly disagree with this path. Not only does it require the Bluetooth connection to be in a specific mode called SSP debug mode, which is not necessarily embedded in all Bluetooth devices, but also the BPA 600 device itself is so outdated that it is so challenging to receive help from the manufactured company when there is a problem within the device.

With these conclusions, the current sub-group suggests that future VIP groups avoid using the BPA 600 solely and instead focus more on the actual implementation of NFC on OOB pairing in Bluetooth. Only once this is attempted, the BPA 600 may be useful to check and analyze potential errors or validity of the OOB pairing.

X. References

[1] D. Perry, L. Wilson and V. Mooney, “Bluetooth Out of Band security”, VIP Secure Hardware, Georgia Tech, Apr. 2020

[2] D. Perry, L. Wilson and V. Mooney, “Bluetooth Out of Band security”, VIP Secure Hardware, Georgia Tech, Nov. 2020

[3] ″Bluetooth Core Specification 5.0”, Bluetooth SIG., Dec. 2016, Accessed 03/08/2023:

https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=421043

[4]″What is Bluetooth Address (BT_ADDR)”, MacAddressChanger, Accessed 03/08/2023:

https://macaddresschanger.com/what-is-bluetooth-address-BD_ADDR

[5] D. Filizzola, S. Fraser and N. Samsonau, “Security Analysis of Bluetooth Technology“, MIT, Accessed 03/08/2023:

https://courses.csail.mit.edu/6.857/2018/project/Filizzola-Fraser-Samsonau-Bluetooth.pdf

[6] ″Bluetooth 5.3 Feature Enhancements Update”, Bluetooth SIG., Jun. 2021

[7] ″Full LE Audio Specification Crosses the Finish Line”, Laird, Jul. 2022, Accessed 03/08/2023:

https://www.lairdconnect.com/resources/blog/full-le-audio-specification-crosses-finish-line

[8] A. Alfarjat, J. Hanumanthappa and H. Hamatta,“Implementation of Bluetooth Secure Simple Pairing using Elliptic Curve Cryptography,” Mar. 2021, Accessed 03/08/2023:

http://paper.ijcsns.org/07_book/202103/20210309.pdf

[9] ″Bluetooth Core Specification v5.0,” Bluetooth SIG., Dec. 2016, Accessed 03/08/2023:

https://www.bluetooth.com/specifications/specs/core-specification-5-0/

[10] "Bluetooth LE Security Study Guide", Bluetooth SIG.

[11] "Bluetooth Pairing Part 1 – Pairing Feature Exchange", Bluetooth SIG., Mar. 2016, Accssed 03/08/2023:

https://www.bluetooth.com/blog/bluetooth-pairing-part-1-pairing-feature-exchange/

[12] J. Fontejon, J. Liu, Y. Vunnam and V. Mooney, "Bluetooth NFC", VIP Secure Hardware, Georgia Tech, Dec. 2022

[13] Y. Shaked and A. Wool, "Cracking the Bluetooth PIN*", 2005, Accessed 03/08/2023:

https://www.usenix.org/legacy/event/mobisys05/tech/full_papers/shaked/shaked.pdf

[14] J. Pearson and V. Mooney, "Classic Bluetooth", VIP Secure Hardware, Georgia Tech, Dec. 2020

[15] "OOB Example", Silicon Labs, Accessed 03/08/2023:

https://docs.silabs.com/bluetooth/2.13/code-examples/stack-features/security/oob-example

[16] "Bluetooth Hacking | How to Protect Yourself", TSS., Aug. 2018, Accessed 03/08/2023:

http://www.texassecurity.net/2018/bluetooth-hacking-protect/

[17] "3. Pairing and bonding", Renesas, Accessed 03/08/2023:

http://lpccs-docs.renesas.com/Tutorial-DA145x-BLE-Security/pairing_and_bonding.html

[18] "Man in the Middle attack in Diffie-Hellman Key Exchange", Geeksforgeeks, Jul. 2022, Accessed 03/08/2023:

https://www.geeksforgeeks.org/man-in-the-middle-attack-in-diffie-hellman-key-exchange/

[19] "BPA® 600 Dual Mode Bluetooth® Protocol Analyzer", Teledyne Lecroy, Accessed 03/08/2023:

https://fte.com/products/bpa600.aspx

[20] "Bluetooth Core Specification 5.4", Bluetooth SIG., Jan. 2023, Accessed 04/04/2023:

https://www.bluetooth.com/specifications/specs/core-specification-5-4/

[21] "Bluetooth About Us", Bluetooth SIG., Accessed 04/04/2023:

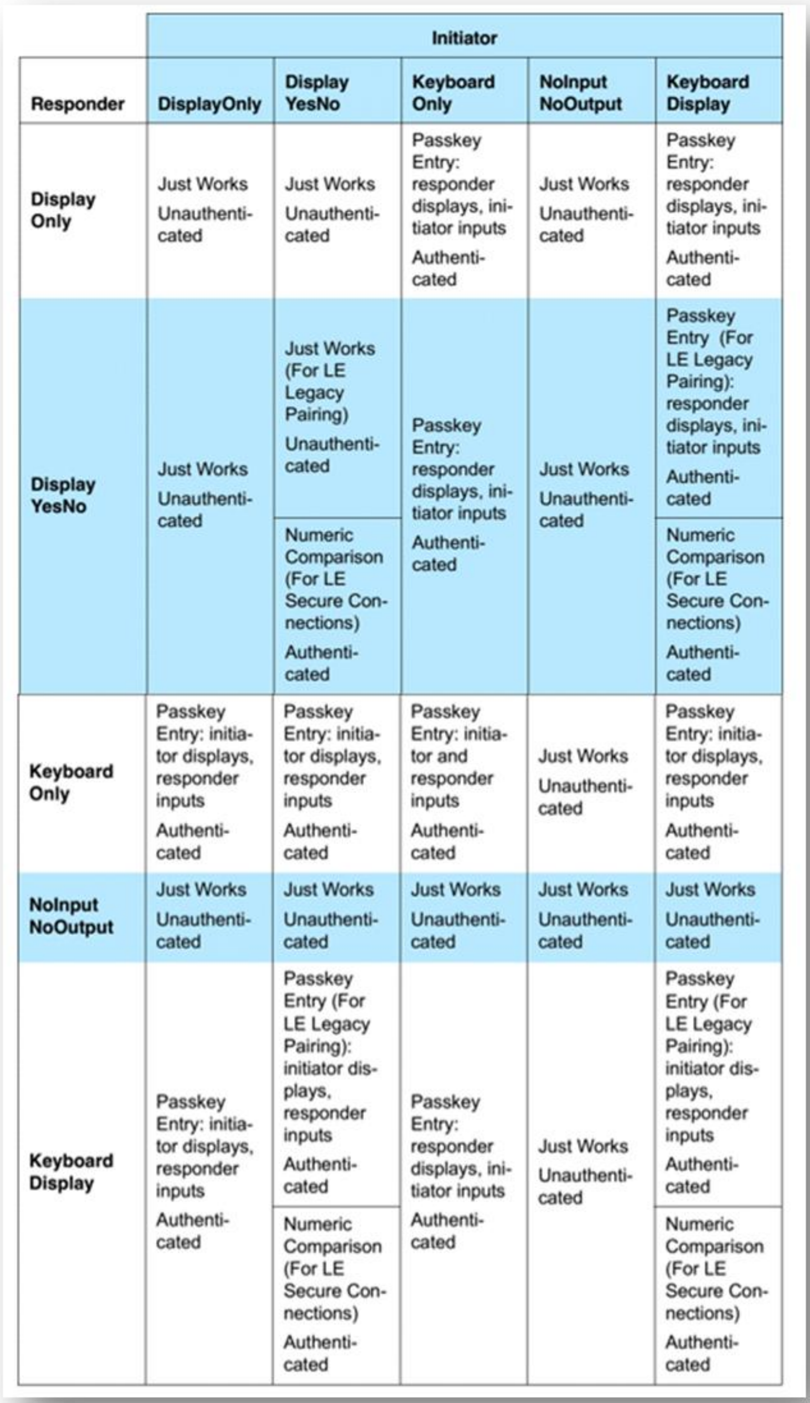
https://www.bluetooth.com/about-us/

[22] "Bluetooth Member Directory", Bluetooth SIG., Accessed 04/04/2023:

https://www.bluetooth.com/develop-with-bluetooth/join/member-directory/

XI. Appendix

X.A Pairing Methods & IO Cap.



	Initiator				
Responder	**DisplayOnly**	**Display YesNo**	**Keyboard Only**	**NoInput NoOutput**	**Keyboard Display**
Display Only	Just Works Unauthenti-cated	Just Works Unauthenti-cated	Passkey Entry: responder displays, ini-tiator inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry: responder displays, ini-tiator inputs Authenti-cated
Display YesNo	Just Works Unauthenti-cated	Just Works (For LE Legacy Pairing) Unauthenti-cated Numeric Comparison (For LE Secure Con-nections) Authenti-cated	Passkey Entry: responder displays, ini-tiator inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry (For LE Legacy Pairing): responder displays, ini-tiator inputs Authenti-cated Numeric Comparison (For LE Secure Con-nections) Authenti-cated
Keyboard Only	Passkey Entry: initia-tor displays, responder inputs Authenti-cated	Passkey Entry: initia-tor displays, responder inputs Authenti-cated	Passkey Entry: initia-tor and responder inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry: initia-tor displays, responder inputs Authenti-cated
NoInput NoOutput	Just Works Unauthenti-cated	Just Works Unauthenti-cated	Just Works Unauthenti-cated	Just Works Unauthenti-cated	Just Works Unauthenti-cated
Keyboard Display	Passkey Entry: initia-tor displays, responder inputs Authenti-cated	Passkey Entry (For LE Legacy Pairing): initiator dis-plays, responder inputs Authenti-cated Numeric Comparison (For LE Secure Con-nections) Authenti-cated	Passkey Entry: responder displays, ini-tiator inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry (For LE Legacy Pairing): initiator dis-plays, responder inputs Authenti-cated Numeric Comparison (For LE Secure Con-nections) Authenti-cated

Table 4. A table describing the final method of authentication during Bluetooth pairing, based on the IO Cap. of both devices.