Use of Out of Band in Bluetooth

04/05/2023

^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

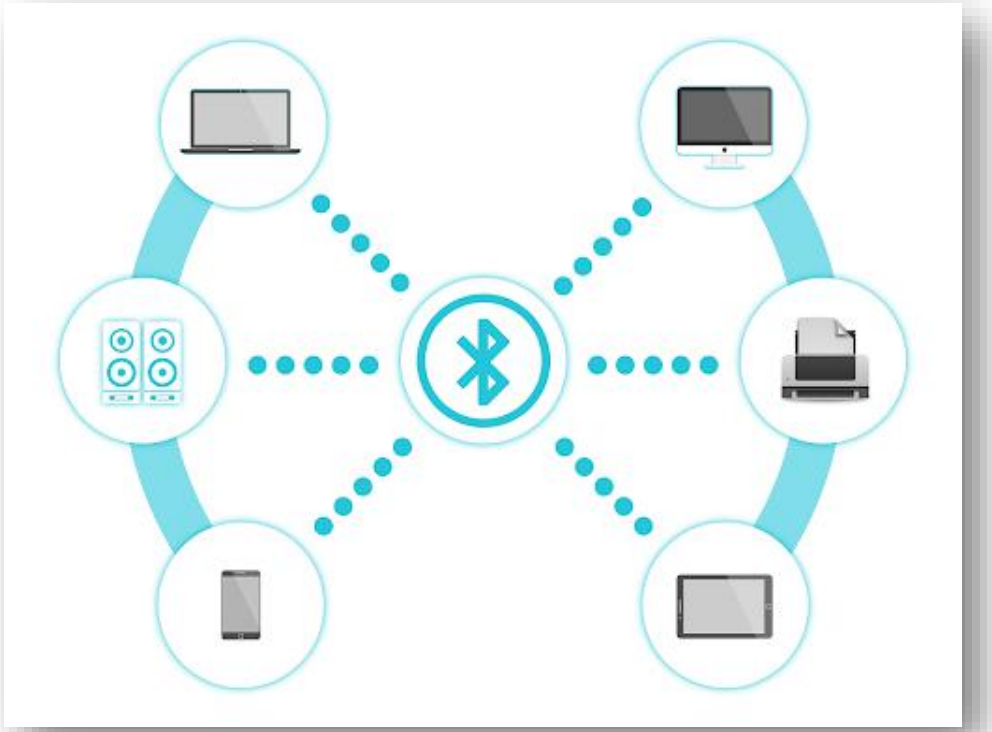
~School of Cybersecurity and Privacy, College of Computing

Georgia Tech

Contents

Introduction

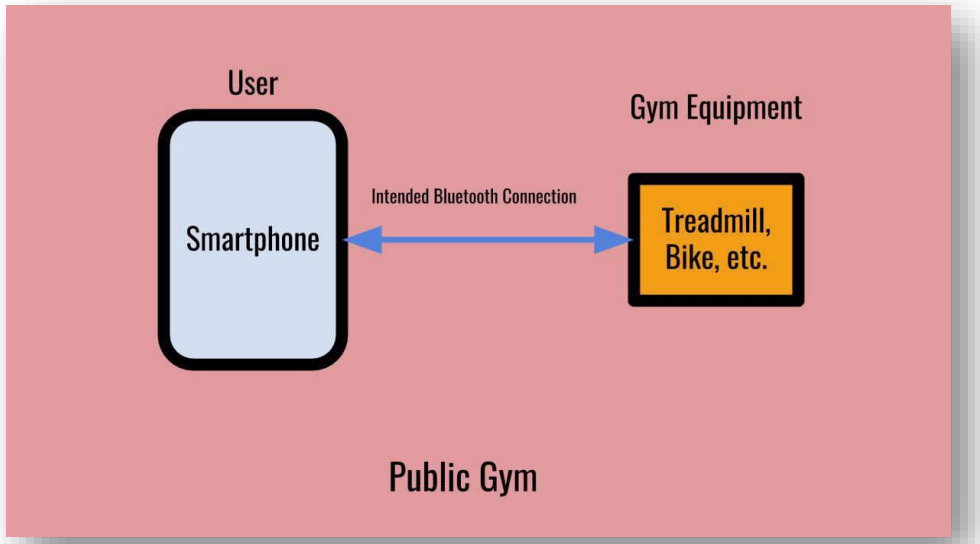
- Since its commercial introduction in 1999, Bluetooth has become one of the most popular methods of wireless communication [2]
- As the technology becomes more widely-used the data being sent across it is a valuable target for hackers [2]
- While some vulnerabilities have been directly addressed in iterations of the Bluetooth standard, Man in the Middle (MitM) attack still remains

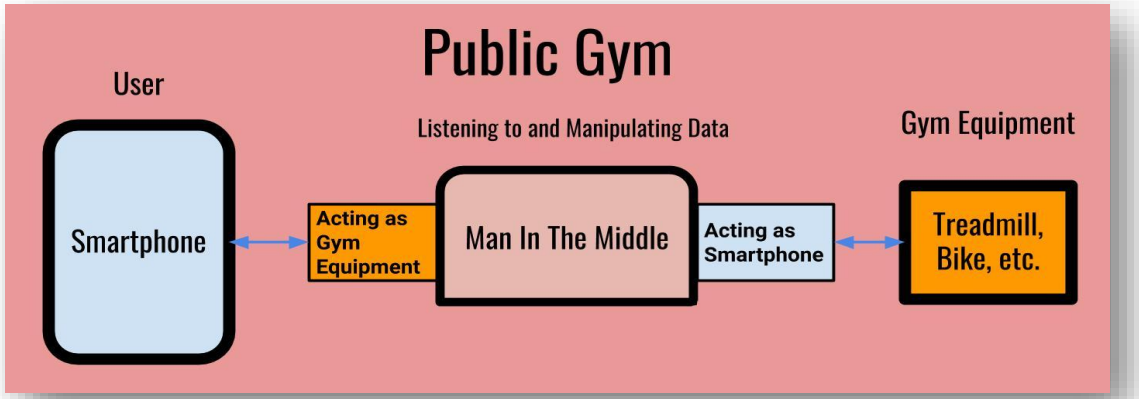


[16]

Scenario Overview

- In a gym, patrons may connect their devices to public equipment using Bluetooth, which will share personal information
  - Constantly mass amounts of pairing process ongoing
- This connection is insecure
  - Pairing through Bluetooth Just Works allows MitM attacks
  - Attacker could easily operate hidden, due to the range of commonly used Bluetooth (~10m)





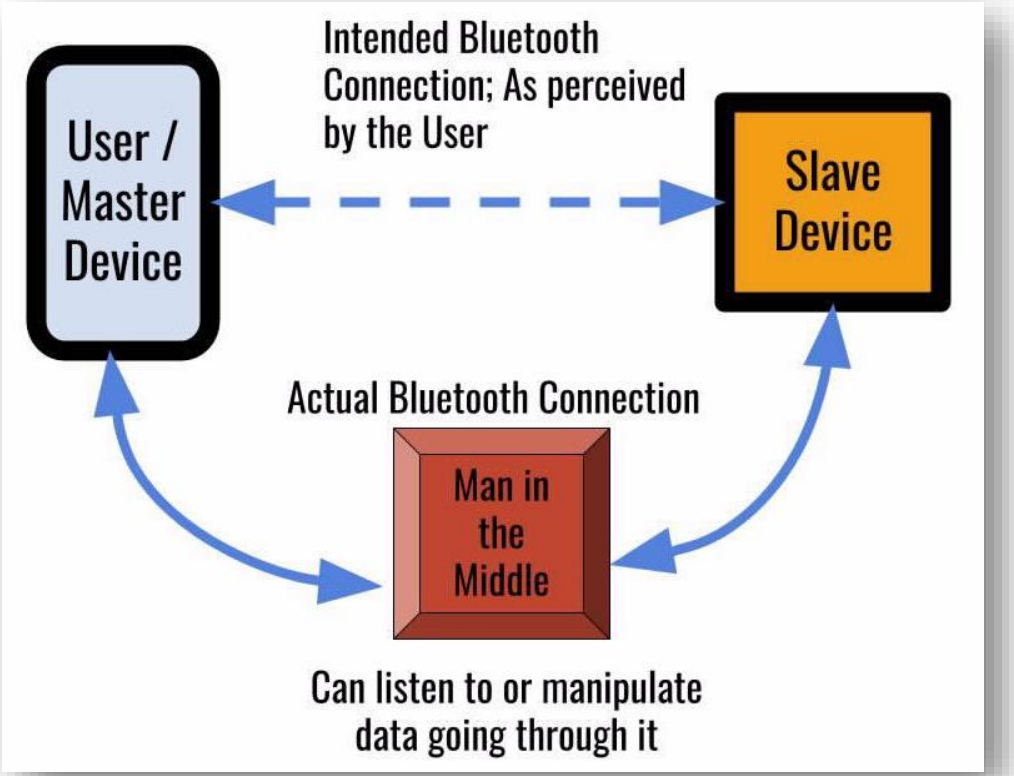
[1]

Georgia Tech

Proposed Solution

- Out-of-Band pairing is a feature of Bluetooth standards that allows pairing through a non-Bluetooth medium
  - Specifically for initial connection
- Among options, use Near Field Communication (NFC) [2]
  - Communication medium requiring close proximity
  - Difficult to compromise

- This research aims to reduce the chance of MitM attacks, support Bluetooth methods such as OOB pairing through NFC, and prevent low entropy keys



[2]

Georgia Tech

Contents

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

Terminology

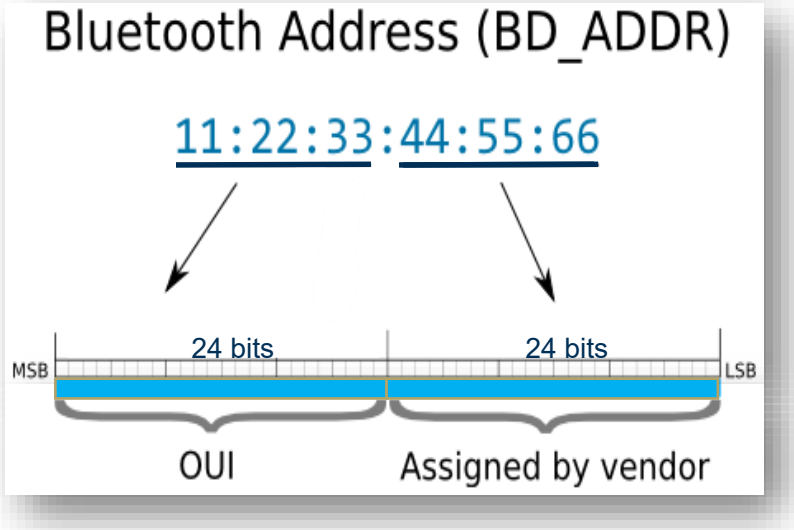
- **OOB**: Out-of-Band Pairing, a way to connect Bluetooth devices using a medium other than Bluetooth

Contents

Bluetooth

- Wireless communications technology which operates at the 2.4GHz frequency band [3]
  - Most equipped with range of 10 m or 33 ft (Class 2)
  - There is a trade-off between range and power consumption within Bluetooth radios.
- Allows for connections between electronic devices, sending relatively small amounts of data over short distances [4] [5]
- Devices have a 48-bit address, also known as the MAC address [5]
  - First significant 24 bits potentially unique to manufacturer, Organizationally Unique Identifier (OUI)
  - Last significant 24 bits intended to be unique to the device

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



Bluetooth Special Interest Group (SIG) [21] [22]

- Standards organization over the development of Bluetooth standards and licensing of the Bluetooth technologies and trademarks to manufacturers (1998~)
- Board of directors
  - composed of one representative from each Promoter member company plus up to four Associate Member Directors (AMD)
  - Microsoft, Intel, Toshiba, Telink Semiconductor, Nokia, Google, Ericsson AB, Motorola, Apple
- Membership:
  - Promoter members, Associate members, Adopter members
  - Promoters: considerable influence over strategic and technological directions of Bluetooth
- Structure:
  - Study groups, Expert groups, Working groups, and Committees
  - Working groups: develop new Bluetooth specifications and enhance adopted specifications, being responsible for majority of published standards and specifications (participation restricted to Promoter members and Associate members)

Current Standards

- 07.2021 – Bluetooth SIG releases **Bluetooth 5.3** [6]
  - Enhancement in data transmission during periodic advertising
  - Enhancement over encryption key size negotiations
  - Allow specific channel selection when adaptive frequency hopping
  - Enhancement over sudden change between power saving mode and higher bandwidth usage
- Potential release of Bluetooth 5.4 [20]
  - Advertising coding selection, encrypted advertising data, periodic advertising with responses
  - No particular security update regarding pairing to be expected, therefore vulnerability of MitM attack still remains

Pairing Process

- To connect, there is a multistep process for the devices [1]:
  1. One device must run an inquiry to discover the other and any devices listening for this inquiry (discoverable) will respond with their address.
  2. Once the desired address is found, pairing request and response packets are sent between the two devices
  3. Both devices will connect once all the information in these packets are received, creating 128-bit pairing key to authenticate connections [17]

- Once connected, they can either be actively sending data or in a sleep mode (LE) until prompted to wake up
- Initially connected devices also can store each other's information in memory, and automatically connect when discoverable in range

Pairing methods

- Legacy Pairing [3]
  - Use simple process of exchanging data to derive a symmetric key with which to encrypt the data transmission during the key distribution phase
  - Not FIPS approved
  - Just Works, Passkey Entry, and Out of Band
- Secure Simple Pairing [3]
  - Introduced in Bluetooth 2.1
  - Use FIPS approved algorithms (4 -> 6 digit) in addition to Legacy Pairing
  - Just Works, Numeric Comparison (not the case for LE SSP), Passkey Entry, and Out of Band
- Secure Connection [3]
  - Introduced in Bluetooth 4.1 to BR/EDR (in LE Bluetooth 4.2)
  - Uses elliptic curve public key cryptography to allow a symmetric key to be derived. That key is then used to derive the link key to be used during Bluetooth data transmission
  - Just Works, Numeric Comparison, Passkey Entry, and Out of Band

Pairing methods

- Except for an additional pairing method of Numeric Comparison, both methods share similarity of transmitting and receiving data for creating the link key (STK, LTK) publicly
  - For Legacy Pairing, Short Term Key (STK) made from the pairing methods is used for creating the Link Key for future connections
  - For connection after the initial pairing, Long Term Key (LTK) is used as the link key
- Securing the process of creating the link key (STK, LTK) is first priority

Pairing Request and Response Packet [11]

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)

- **Code**: indicates either pairing request or pairing response
- **IO Cap.**: I/O Capabilities, indicates which abilities the device have on communication
- **OOB Data Flag**: Out-of-Band Data Flag, indicates external means of communication
- **AuthReq**: Authentication Request, indicates the want for options such as MitM protection, generation of long-term key, LE Secure Connection []
- **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 Long Term Key (LTK)

IO Cap. and Pairing Methods

- Show what capabilities device have for authentication [11]
- Input-wise
  - Keyboard
  - Simple Yes/No button
- Output-wise
  - Display

- Comparing IO Cap. values from both devices, decide which paring method to take
  - Just Works, Passkey Entry, Numeric Comparison, OOB

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

Contents

Out of Band (OOB) Pairing

- Feature described in Bluetooth standard that allows the pairing process to occur over a non-Bluetooth medium [12]
  - Discovery information is exchanged through this medium
    - Bluetooth Address
    - Temporary Key (TK) for authentication
- TK is later used in creation of the Short Term Key
- Both devices must set their OOB flag to initiate pairing
  - 1 byte flag
  - If not set, OOB pairing not initiated

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

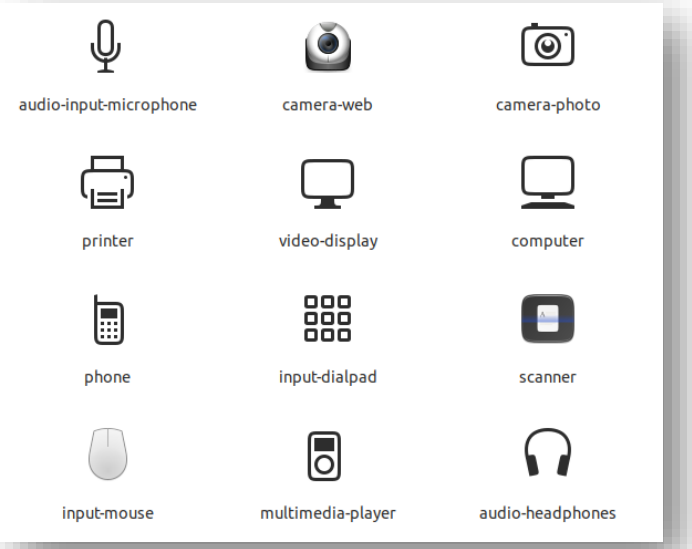
Contents

Georgia Tech

Just Works

- When IO Cap indicates Passkey entry, Numeric comparison nor OOB can be used
- Entropy is non existent
- “Just Works pairing offers no authentication and, hence, no protection against MitM attacks.” [10]
- While Passkey entry pairing method is also vulnerable to MitM attack with bruteforce, LE Secure Connection has made it difficult through rigorous checking procedures [13]

- Therefore, Just Works remains as the single most vulnerable pairing method



Georgia Tech

MitM vulnerability with Just Works – Legacy Pairing [10]

1. Authentication step
   - Create TK (Temporary key) of 128-bit, always 0
   - Central and Peripheral create and exchange $Rand_c$, $Rand_p$ and uses it to derive C_c, C_p through a function

   - C_c = c(TK, $Rand_c$, Pairing Request command, Pairing Response command, address $type_c$, $address_c$, address $type_p$, $address_p$)
   - C_c, C_p is then exchanged to check authenticity

2. STK creation
   - Create STK using TK, $Rand_c$, $Rand_p$ through function

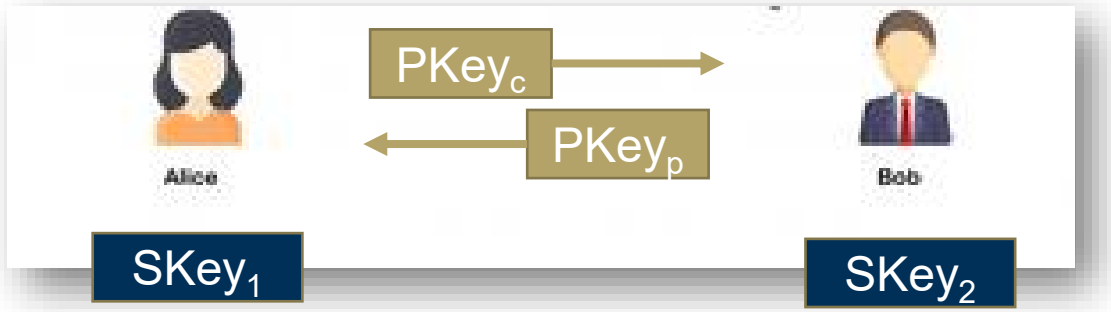
   - STK = s(TK, $Rand_c$, $Rand_p$)
   - STK is used to create the link key for the initial session, and in most cases, LTK is created to be used in future link key creations (bonding)

MitM vulnerability with Just Works – Secure Connection [10]

1. Creating the DHKey (Deffie Hellman key) – Secure Connection
   - Both central and peripheral devices create $SKey_c$, $SKey_p$ (Secure Key), and exchange $PKey_c$, $PKey_p$ (Public Key)
   - DHKey = P256(SK_c, PK_p)

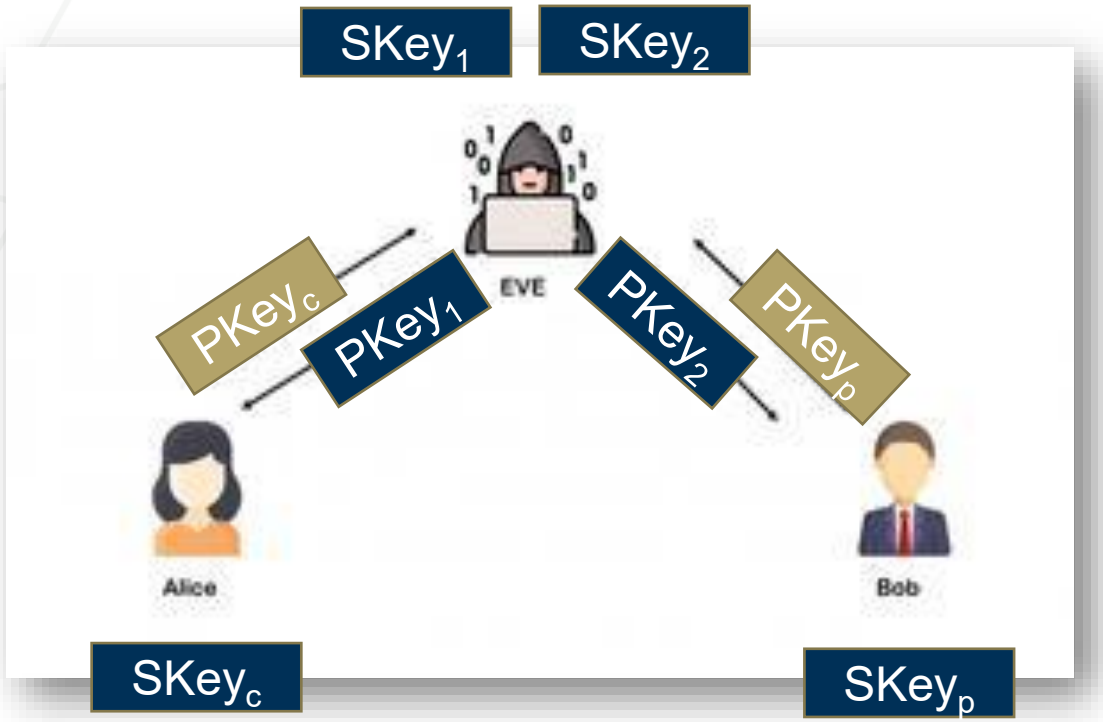
- MitM scenario
  - Attacker creates $SKey_1$, $SKey_2$, $PKey_1$, $PKey_2$ and ends up creating two separate keys $DHKey_1$, $DHKey_2$

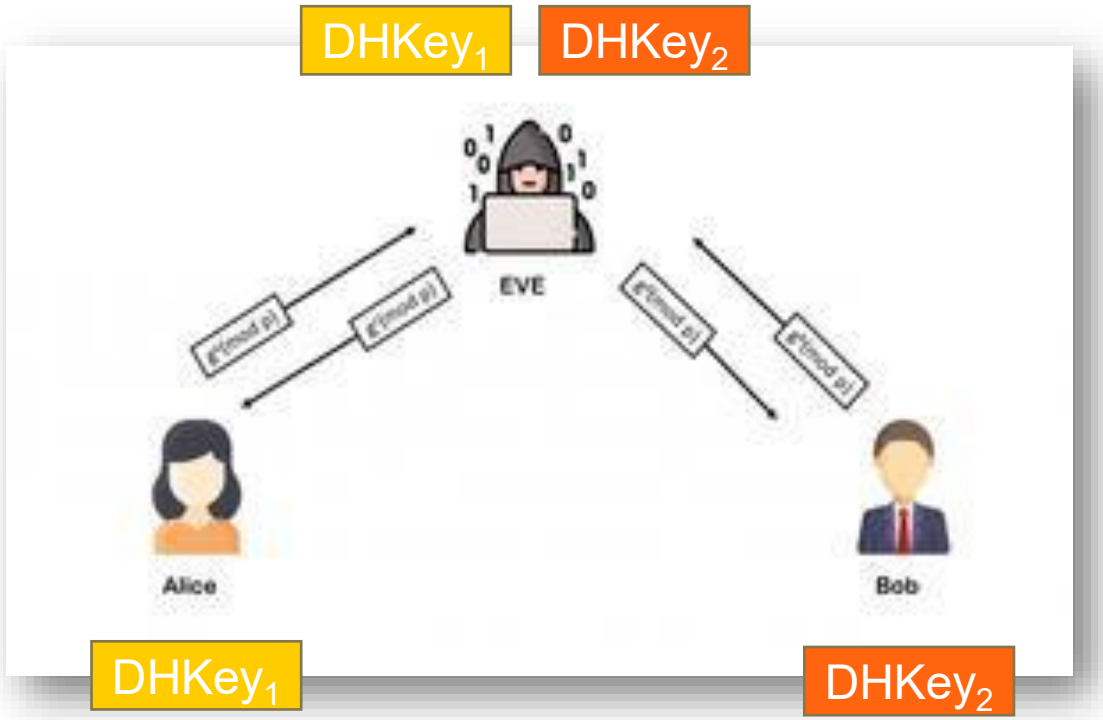
- Vulnerability by Deffie Hellman in authentication process



[18]

--[Simpler diagram]





DHKey = P256($SKey_c$, PK_1)

DHKey = P256(SK_p, PK_2)

Georgia Tech

MitM vulnerability with Just Works – Secure Connection [10]

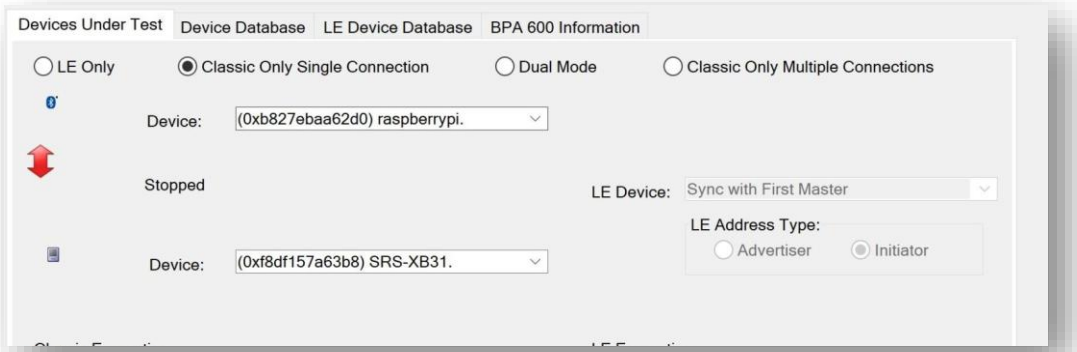
2. Authentication step - C
   - Central and Peripheral create and exchange N_c, N_p, pseudo-random 128-bit number or nonce, and uses it to derive C_c, C_p through a function
   - $C_c = f1(PK_c, PK_p, N_c, 0)$
   - C_c, C_p is then exchanged to check authenticity
3. Authentication step – E and LTK creation
   - Create Mackey and LTK using DHKey, N_c, N_p, and Bluetooth addresses through a function
   - $MacKey \,||\, LTK = f2(DHKey, N_c, N_p, address_c, address_p)$
   - Mackey is used create and exchange E_c, E_p (AES-CMAC) to check authenticity
   - Once all steps are passed, LTK is used to create the link key and saved for future connections

Contents

BPA 600

- Bluetooth sniffer [14]
  - Definition: A chipset hardware device that intercepts, captures, and analyzes Bluetooth signals and their data by occupying 2.4GHz ISM band
  - Purpose: To capture, decode, and analyze Bluetooth packets in real time

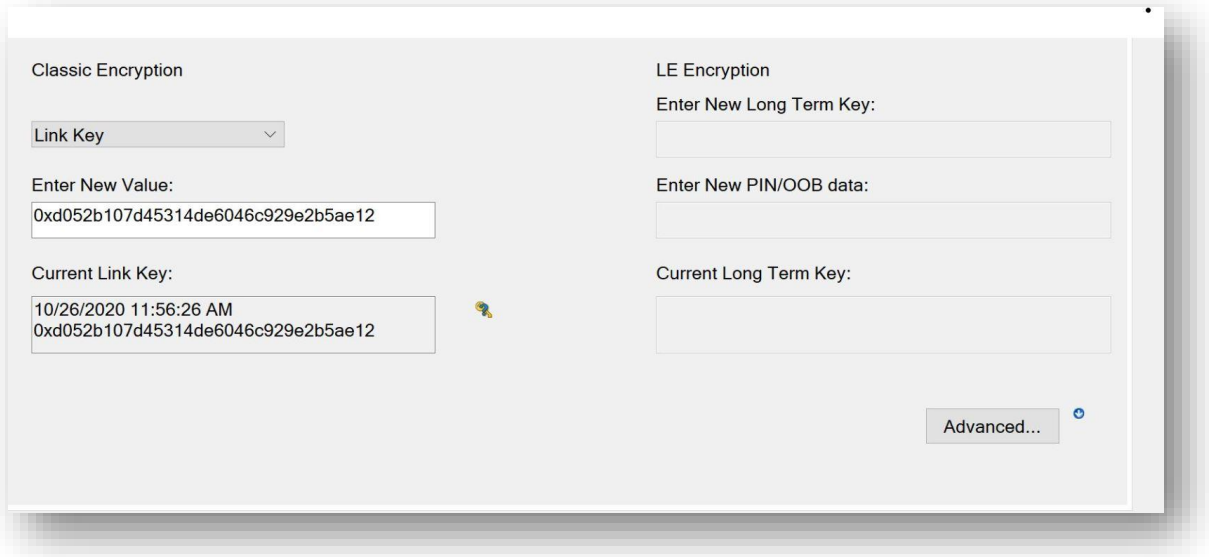
- Hardware: BPA 600 [19]
  - Functionality:
    - operable on devices that are up to and including Bluetooth 4.2 specification.
    - USB powered device that uses two detachable antennas to sniff both LE and Classic Bluetooth connections.
    - 4 different sniff modes: LE, Classic, Dual Mode (LE and Classic), and Classic-only Multiple Connections
    - allows the user to input keys directly or in some cases, the software can generate the keys



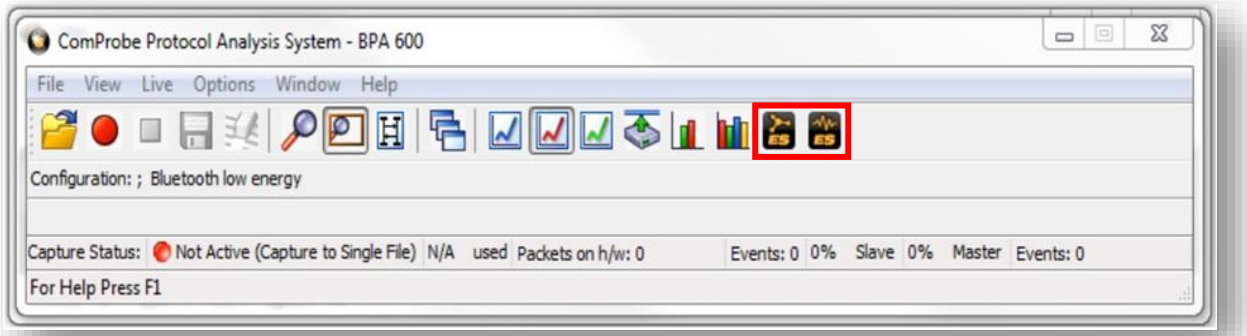
Georgia Tech

BPA 600

- Decryption Functionality[19]
  - Classic connections: can input the PIN code used during pairing (Passkey entry) or the Link Key directly (manually search log file of Bluetooth device)
  - LE connections: can input the LTK directly or input the OOB data (both require manually search log file of Bluetooth device)



- Audio Expert System
  - Allows to directly decrypt audio packets sent through Bluetooth which has been captured



Georgia Tech

Contents

Experiments

- Experiment procedure:
  1. Enable connection between two Bluetooth devices
  2. Capture packets during Bluetooth communication (audio or file transfer) with BPA 600
  3. Through BPA 600's software, analyze packets and decrypt message

- Bluetooth devices for connection
  - 2 * Raspberry pi (with Plugable USB 2.0 Bluetooth Adapter)
    - Plugable USB 2.0: Supports up to Bluetooth 4.0
    - MAC Address: 5C:F3:70:87:FD:A2
    - MAC Address: 5C:F3:70:87:FD:A7
  - Android (Samsung S10 5G)
    - Personal device prepared for backup
    - MAC Address: 64:7B:CE:60:01:31
  - Sound device (Samsung Galaxy Buds 2 Pro)
    - Personal device prepared for backup
    - MAC Address: B0:A4:6A:77:BB:20

Experiments

- Experiment 1:
  - Connection between Raspberry Pi – Raspberry Pi
  - While Bluetooth UI exists for the Rasp OS to easily connect to Bluetooth, connected through “bluetoothctl” for altering specific conditions (Controlling IO Cap for the initial advertising packet)

  - Steps (enable certain IO Cap):
    1. Open a terminal. Run “bluetoothctl agent off”
    2. Run “bluetooth agent NoInputNoOutput”
    3. Run “bluetooth agent on”
  - Steps (Pair 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]”
- Result:
  - Despite giving the correct message that agent was on, IO Cap were not set correctly, and connection between both Raspberry Pi devices were not made

Experiments

- Experiment 2:
  - Connection between Raspberry Pi – Android
  - Goal is to get a Bluetooth connection, and send files for the BPA 600 to capture and decrypt

  - Steps (Pair 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)

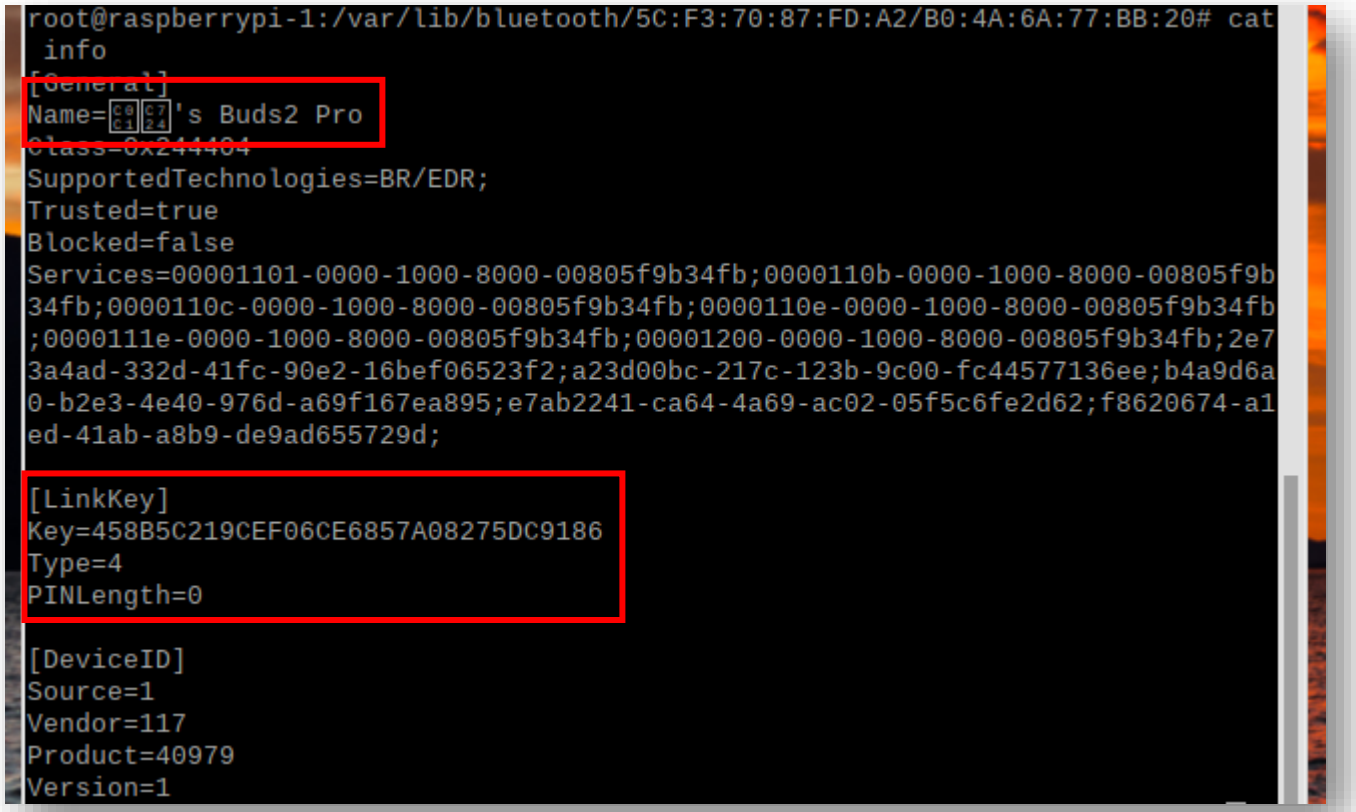
- Result:
  - Connection was successfully made, but individually downloading and setting up ways for either Raspberry Pi or the Android device to send certain data (ie. File) were mostly outdated or not working

Experiments

- Experiment 3:
  - Connection between Raspberry Pi – Sound device
  - Goal is to get a Bluetooth connection, and send sound data for the BPA 600 to capture and decrypt

  - Steps (Pair 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)
  - Steps (Derive 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

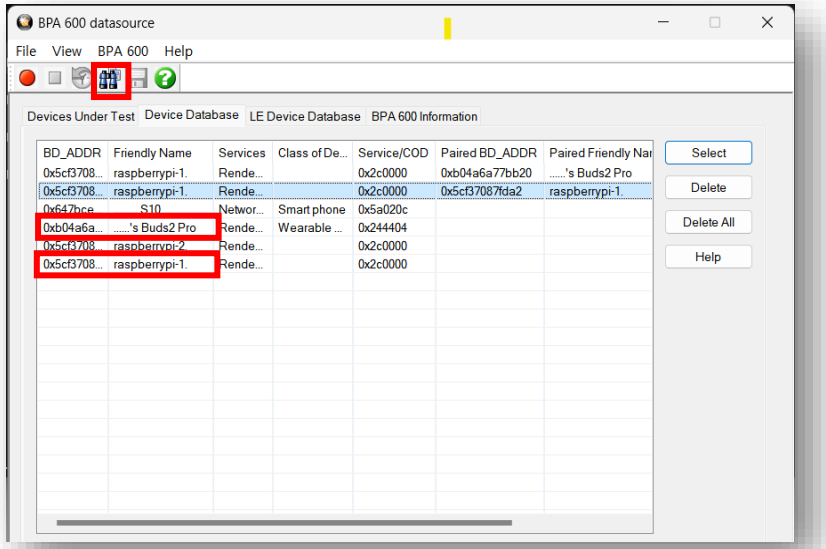
Experiments

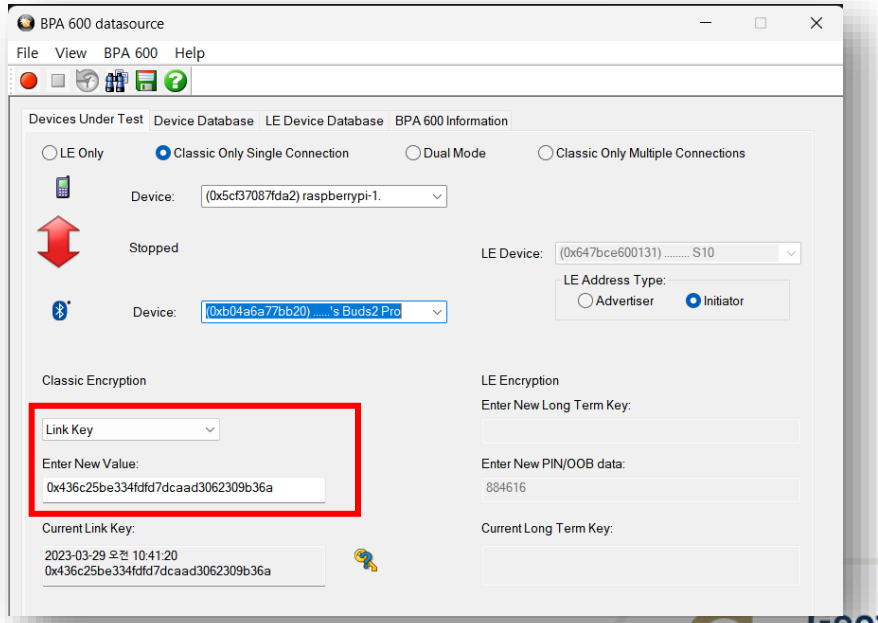


Georgia Tech

Experiments

- Experiment 3:
  - Steps (Prepare and capture):
    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
  - Steps (Decrypt):
    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

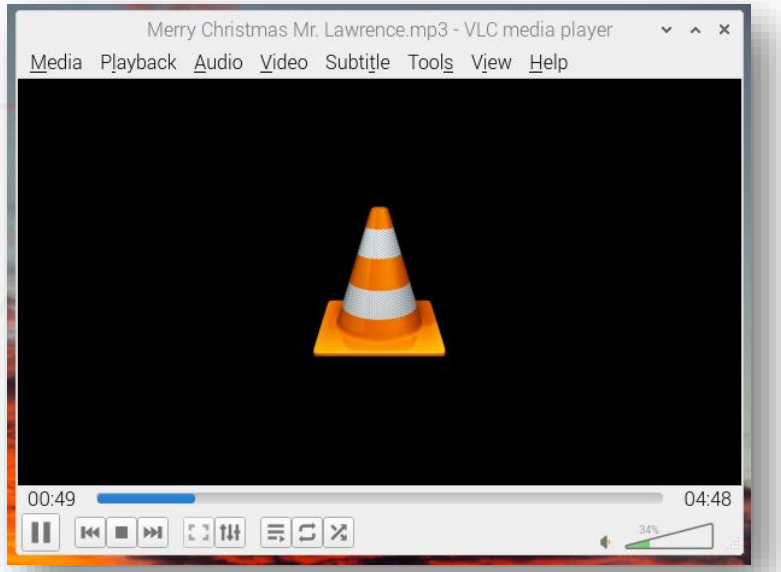


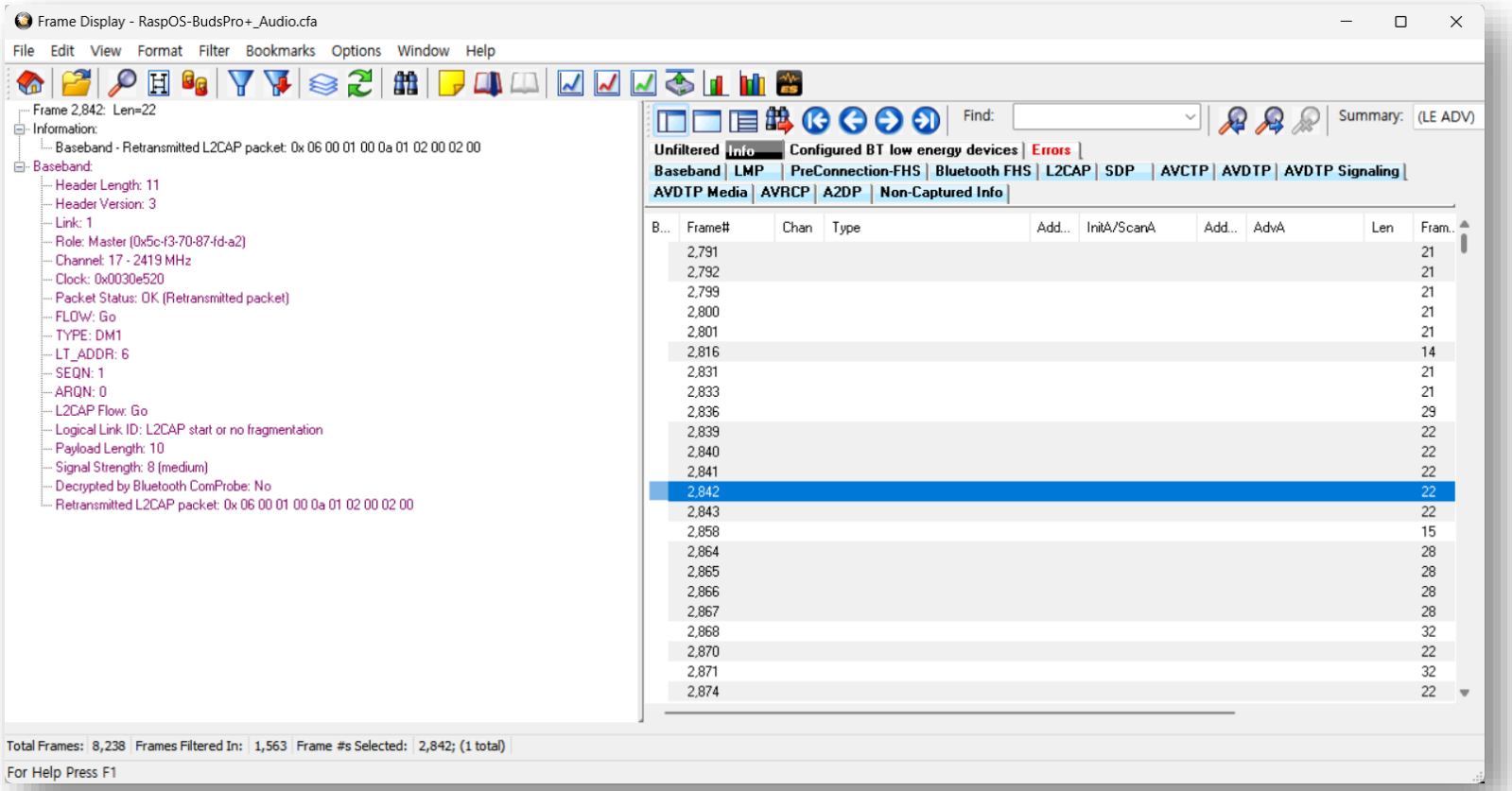


Georgia Tech

Contents

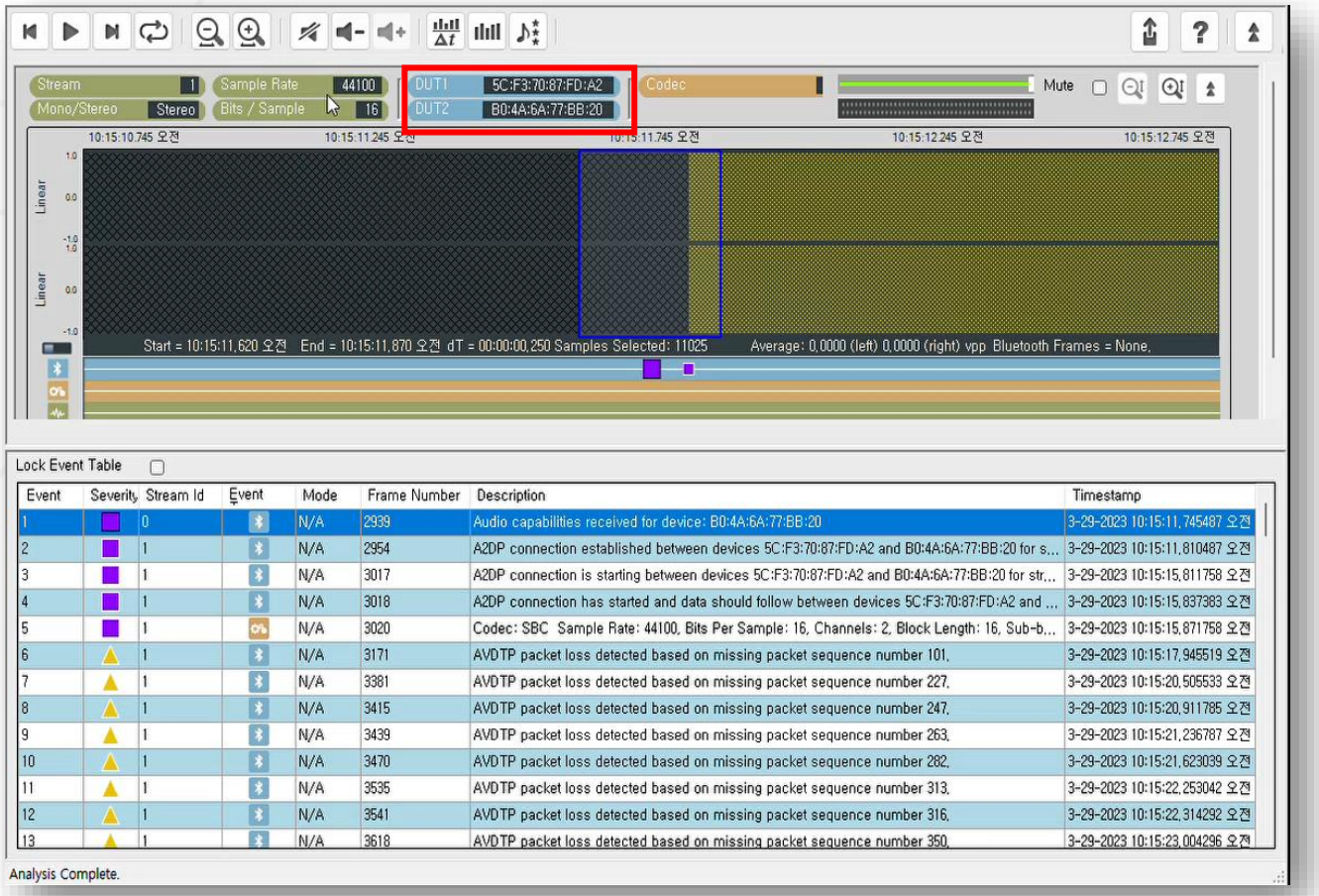
Results





Georgia Tech

Results



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

- Takeaway: ensure current understanding of Bluetooth connection
  - Legacy pairing + SSP – Secure Connection

Georgia Tech

Contents

Future Work

- Previous research focused on BPA 600's ability to decrypt connection without providing information to the encryption process (Link key, LTK, Pin code) when connected through "SSP debug mode" [14]
  - manufactures not required to include SSP Debug mode, so could be considered less relevant
- Usage of BPA 600 only remains as to check basic understanding of Bluetooth connection
  - Out-dated device which is already hard to receive help from manufacture, and using commercial devices only allow to check one's understandings
- Future research should preferably avoid using BPA 600 and instead focus more on researching and implementing OOB
  - Once OOB is successfully implemented via NFC, the device may come in handy to check connections through OOB pairing

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/

References

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/

References

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/

Appendix– 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, initiator inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry: responder displays, initiator inputs Authenti-cated
Display YesNo	Just Works Unauthenti-cated	Just Works (For LE Legacy Pairing) Unauthenti-cated	Passkey Entry: responder displays, initiator inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry (For LE Legacy Pairing): responder displays, initiator inputs Authenti-cated
		Numeric Comparison (For LE Secure Connections) Authenti-cated			Numeric Comparison (For LE Secure Connections) Authenti-cated
Keyboard Only	Passkey Entry: initiator displays, responder inputs Authenti-cated	Passkey Entry: initiator displays, responder inputs Authenti-cated	Passkey Entry: initiator and responder inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry: initiator 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: initiator displays, responder inputs Authenti-cated	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenti-cated	Passkey Entry: responder displays, initiator inputs Authenti-cated	Just Works Unauthenti-cated	Passkey Entry (For LE Legacy Pairing): initiator displays, responder inputs Authenti-cated
		Numeric Comparison (For LE Secure Connections) Authenti-cated			Numeric Comparison (For LE Secure Connections) Authenti-cated

[11]