



# Bluetooth Security

Securing Wireless Networks COMP4337/9337

Never Stand Still

Dr. Hailun Tan  
Postdoc fellow

School of Computer Science and Engineering, UNSW


Hi good afternoon everyone, my name is Hailun and I am a postdoc fellow at CSE in UNSW.

Today I am going to give a lecture about Bluetooth Security.

2

# Roadmap

- Introduction
- Features
- Security Issues
- Case Study – packet sniff
- Conclusions

-

Here is the roadmap of today's talk about Bluetooth security. First, I will give you a brief introduction about the Bluetooth.

Then I will go through the features of Bluetooth. Then we will cover security issues associated with Bluetooth communications.

We will also have a look at one of the Bluetooth vulnerability in Bluetooth Smart (Low Energy) in a case study

about how to hack into a Bluetooth network by launching a packet sniffing attack. Finally we go through the conclusions.

# Introduction

- Open wireless protocol for exchanging data over short distances from fixed and mobile devices, creating personal area network.
- A reliable wireless protocol for voice and data transmission

Bluetooth is basically an open wireless protocol.

I think most of you have used Bluetooth before to exchange data over a very short distance probably up to 5 to 10 meters.

From either fixed or mobile devices such as headphones and Bluetooth speakers etc.

Headphone is for voice transmission and Bluetooth mouse/keyboard transmit data.

# Bluetooth Evolution

- Bluetooth Special Interest Group (SIG)
- Founded in Spring 1998
- By Ericsson, Intel, IBM, Nokia, Toshiba
- Now more than 2,000 organizations have joined the SIG

-




Bluetooth was founded in 1998 and It has a Bluetooth Special Interest Group (SIG) founded by consortium of companies including Ericsson, Intel, IBM etc.

Now more than 2000 industrial organizations have joined this special interest group to evolve the standard and develop Bluetooth related devices and applications.

5

# Roadmap

- Introduction
- **Features**
- Security Issues
- Case Study – packet sniff
- Conclusions

UNSW  
AUSTRALIA  
Cy5PH Laboratory

That is the basic introduction about Bluetooth. Now we come to the features.

## Features

- Bluetooth-enabled devices can automatically locate each other
- Topology is established on a temporary and random basis
- Up to eight Bluetooth devices may be networked together in a master-slave relationship to form a piconet

-



Bluetooth enabled devices that most of you have used, they can automatically locate each other.

When you use one Bluetooth enabled device it searches for other Bluetooth enabled device that are available to connect. These devices form a network in a very ad-hoc way.

If you go to your local shop (such as JB HiFi) to test the quality of a Bluetooth enabled speaker, you can use your Bluetooth enabled mobile device such as iPhone to connect to a Bluetooth speaker and then stream your music to the Bluetooth speaker to play your favorite songs.

The topology is formed on a temporary and random basis. You can thus connect up to 8 Bluetooth devices in a master slave relationship.

This Bluetooth network is called a Piconet. National Institute of Standards and Technology (NIST) defines this network as Wireless Personal Area Network (WPAN).

## Features (Cont.)

- One is master, which controls and sets up the network (piconet)
- Two or more piconet interconnected to form a scatter net
- Only one master for each piconet
- A device can't be masters for two piconet
- The slave of one piconet can be the master of another piconet

In a Piconet, there can be only one master to control and setup the network.  
And two or more Piconet can interconnect to form a Scatter net.

There is only one master allowed for each Piconet.  
A device cannot be a master for two Piconet

However, a slave of one Piconet can be the master of another Piconet.

# Roadmap

8

- Introduction
- Features
- **Security Issues**
- Case Study – packet sniff
- Conclusions

-

Now it comes to the main focus of my talk that is security issues a Bluetooth network can have.



# Security Issues

9

- Authenticity: Are you the device you claim you are?
  - Impersonation
- Confidentiality: Is the exchanged data only available to the intended devices?
  - Packet sniffing
- Authorisation: Are only the intended devices accessing the specified data and control?
  - Prerequisite: authenticity and confidentiality

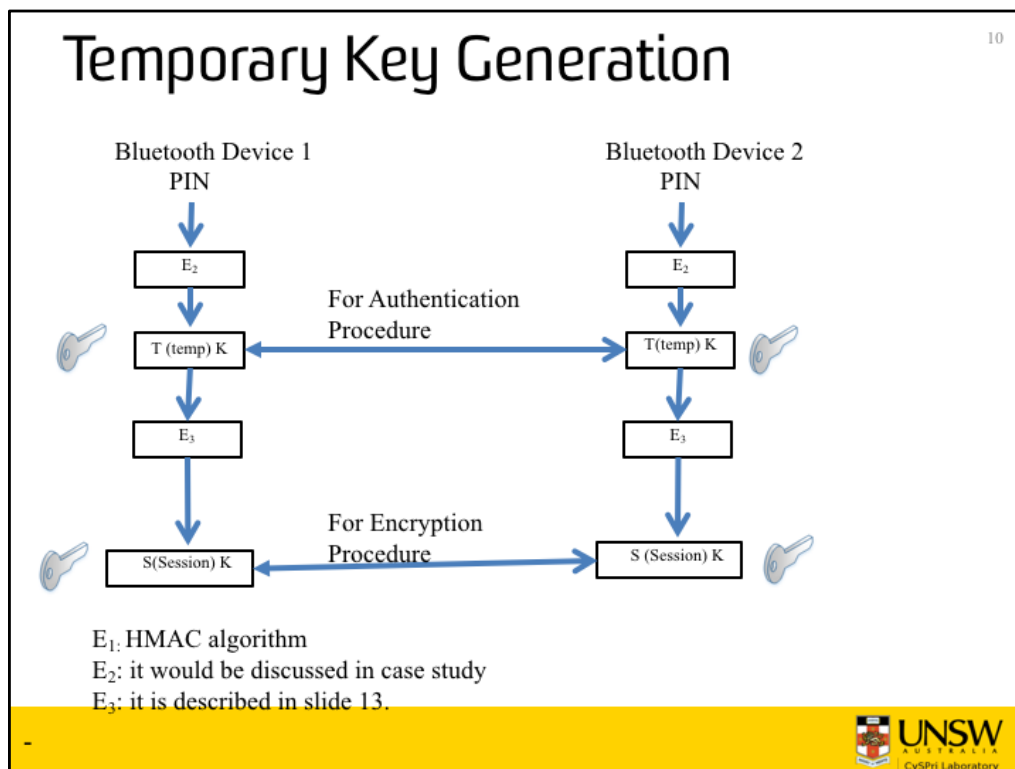


In Bluetooth communications these 3 are the most essential security issues that must be considered.

Authenticity means that: Any Bluetooth device must be able to authenticate themselves to prove/verify who they claim they are. An Impersonation attack is launched pretending to be someone else. If you cannot verify who you are. The protocol should be able to detect this to identify someone posing as another device.

For confidentiality, the data has to be only available to the intended device instead of some other device. Packet sniffing can be launched against this security feature to access data that is being transferred between two devices in an unauthorized manner.

And the third property is Authorization that means only the intended device has access to specified data and control. This combines the above two features. Prove your authenticity first and then exchange data in a secure manner.



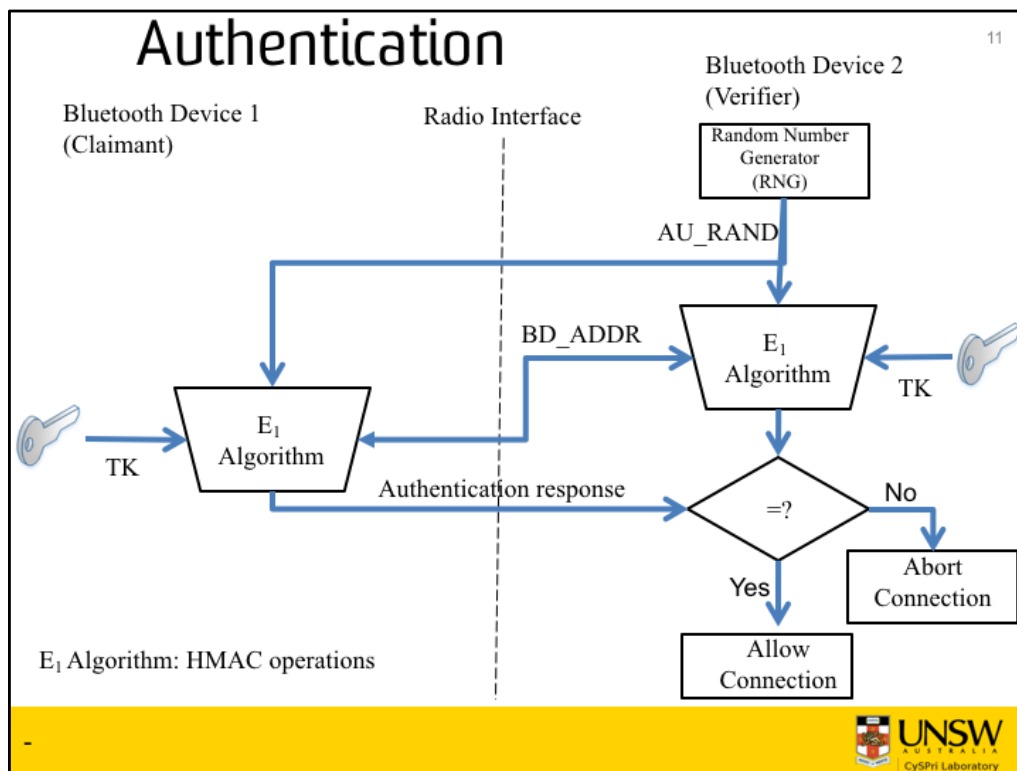
In order to ensure the authenticity, confidentiality and authorisation, one basic component in security is how to establish a key that can be used for encrypting data to ensure confidentiality and at the same time use the key for authorization.

This illustration shows how keys are generated (among two Bluetooth devices) for security operation.

Initially the devices have PIN from the initialisation step. PIN is the default password set in the Bluetooth devices. It is a source of security vulnerability in Bluetooth.

They have algorithms to generate keys at different stages.  $E_2$  is the algorithm to generate the Temporal Key (TK) that is used for authentication process. We will discuss this algorithm in later slides.

Next the devices use another algorithm  $E_3$  to generate another Session key that is subsequently used for encryption while transferring data among the devices. This later key (SK) provides confidentiality and the earlier key (TK) was for authenticity.



For authentication, Bluetooth device 1 is the claimant which has to be authenticated by the verifier Bluetooth device 2.

Step 1: The claimant sends its  $BD\_ADDR$  across to the verifier. The verifier generates a random number ( $AU\_RAND$ , 128 bit) and sends it to the claimant.

Step 2: The claimant uses the  $E_1$  algorithm (Hash-based Message Authentication Code HMAC) to compute an authentication response using its unique 48-bit Bluetooth device address ( $BD\_ADDR$ ), the TK (generated in previous slide), and  $AU\_RAND$  as inputs. The verifier performs the same computation. Only the 32 most significant bits of the  $E_1$  output are used for authentication purposes.

Step 3. The claimant returns the most significant 32 bits of the  $E_1$  output as the authentication response to the verifier.

Step 4. The verifier compares the authentication response from the claimant with the value that it has computed.

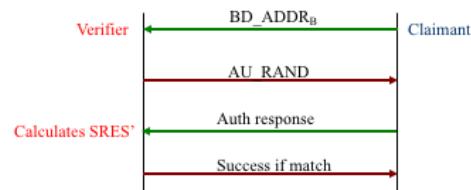
Step 5. If the two 32-bit values are equal, the authentication is considered successful. If the two 32-bit values are not equal, the authentication fails.

Device only gets verified if it possesses the correct TK and can produce the correct authentication response using the same  $E_1$  algorithm.

This is one way authentication (claimant gets verified). Devices can switch roles to get two way authentication.

# Authentication Summary

12



Authentication Process

Parameter	Length	Secrecy parameter
Device Address	48 Bits	Public
Random Challenge	128 Bits	Public
Authentication(Auth) Response	32 Bits	Public
Temporary Key	128 Bits	Secret

If you look at the summary of this authentication process, all the information (parameters) are public except the Temporary Key.

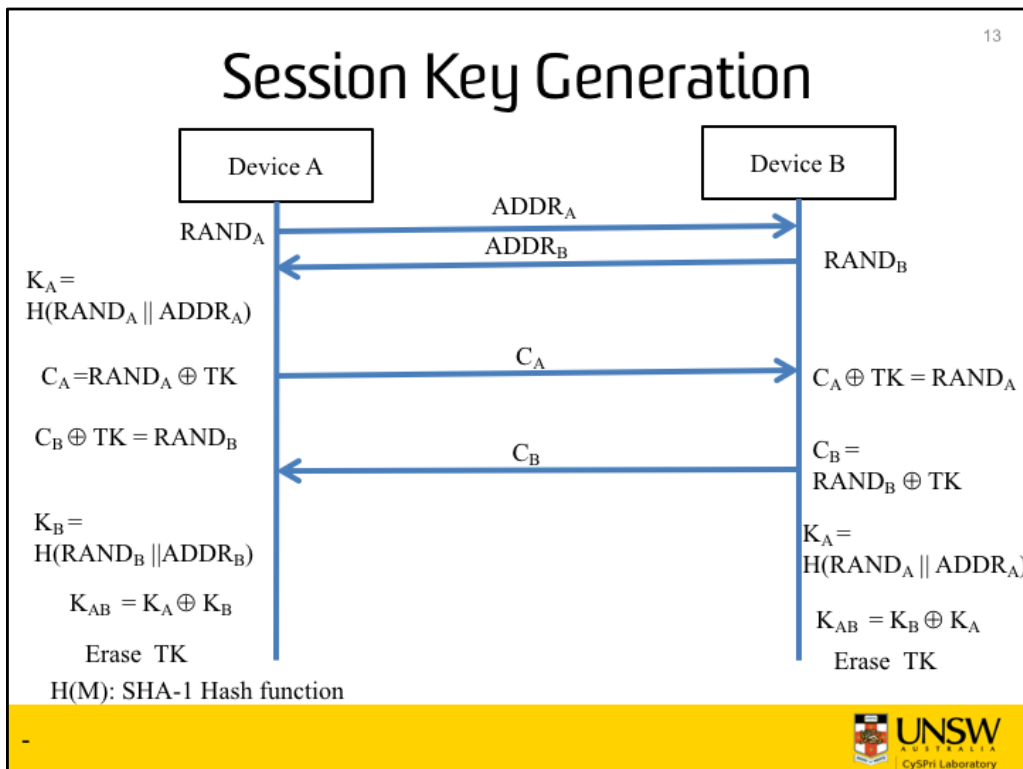
That means these are also known to the third parties (attackers) that can use this information.

If the attacker want to pose as claimant, it has this public information.

The only secret part is the Temporary Key that the attacker does not know.

If the attacker knows the Temporary Key, it can easily impersonate as the claimant.

I will talk about it later how this Temporary Key can be cracked.



This is the process involved in generating the Session Key (SK) that is different from the Temporary Key (TK). SK is used for confidentiality while TK is used for authentication.

If two devices want to establish communication and exchange data, first they have to derive a SK for this session

Step 1: Exchange BT addresses ( $ADDR_A$  and  $ADDR_B$ )

Step 2: Generate a Random number and use one-way hash function on the generated random number and own BT address. Get  $K_A$  at A and  $K_B$  at B.

Hash function is one-way means even if the attacker knows  $K_A$ , it cannot retrieve  $RAND_A$

Step 3: TK already known to both A and B. XOR the generated Random Number with TK to get  $C_A$  (and  $C_B$  at B).

Step 4: Exchange  $C_A$  and  $C_B$

Step 5: Device A XOR  $C_B$  with TK to get  $RAND_B$  and device B XOR  $C_A$  with TK to get  $RAND_A$

Step 6: Device A calculates the Hash using  $RAND_B$  and  $ADDR_B$  to get the  $K_B$  (similarly Device B gets  $K_A$ )

Step 7: XOR  $K_A$  and  $K_B$  to get the final session key  $K_{AB}$ . TK can now be discarded.

This is how we achieve Authenticity and Confidentiality using TK and SK.

Assumption is that TK has already been established to perform this algorithm.

It is reusing TK that is a significant vulnerability as the source of the Temporary Key is not secure making this Session key generation algorithm vulnerable.

## Is Channel Hopping Secure?

- Channel Hopping (Bluetooth Smart only) – Both communication parties would hop to a different wireless channel per packet in a fixed channel hopping increment.
- Adversary could not achieve data by monitoring one wireless channel only.
- We will study its vulnerability soon.

-



Another mechanism to secure wireless networks is called Channel Hopping.

In Channel Hopping, we have multiple channels available within a given frequency for wireless devices to use. Devices select the same channel frequency to communicate.

In Bluetooth Smart protocol, the Bluetooth Special Interest Group (SIG) has specified channel hopping to increase the security.

The reason to do this is that by doing channel hopping to different channels for each data packet, the attacker is not able to fully capture

the whole conversation by monitoring a single channel.

The SIG expected that in order for the attacker to intercept all data packets/conversation, the attacker needs to monitor all channels which increases the cost for the attacker.

This channel hopping basically works with fixed channel hopping increments and we also have vulnerability in this channel hopping mechanism that we will study in the later slides.

# Roadmap

- Introduction
- Features
- Security Issues
- **Case Study – packet sniff**
- Conclusions

-

We have now come to the interesting part which is how the attacker can crack the security mechanism that I mentioned before.

## Case study: Bluetooth Low Energy (BTLE)

- Introduced in Bluetooth 4.0 (2010)
- New modulation and link layer for low power devices
  - Incompatible with classic Bluetooth devices
  - PHY and link layer different (no channel hopping in classic Bluetooth)
  - High-level protocols reused (L2CAP, ATT)

-



Channel hopping is used in Bluetooth Low Energy protocol introduced in 2010. Also called Bluetooth 4.0

Introduce new modulation and link layer for low power devices

It is not compatible with classic Bluetooth devices

High level protocols have been reused



# BTLE applications

- High end smart phones
- Sports/fitness devices
- Door locks
- Upcoming medical devices (e.g., blood glucose monitor)

BTLE is used in high end smart phones , sports/fitness devices and in upcoming medical devices such as blood glucose monitor.

# BTLE Protocol review

GATT (Generic Attribute Profile) – how to discover and provide services based on ATT

ATT (Attribute protocol) – how to discover/read/write attributes on a peer device

L2CAP (Logical Link Control and Adaptation Protocol) – packet segmentation and reassemble

Link Layer

Physical Layer

- We will focus on Link layer and Physical layer security only.
- Other layers are similar to their counterparts in wired networks.

In this topic we will only focus on Link and Physical layer security.

From the security perspective, the higher three layers are similar to their counterparts in the wired networks.

# Physical Layer

19

- Physical layer: channels for hopping (40 available channels in 2.4Ghz)
  - Advertising: 3 channels
  - Data: 37 channels

-



Back to channel hopping:

In Bluetooth low energy communications, there are 40 available channels in 2.4GHz range in total for the communicating parties to use.

Out of these 40, 3 channels are for advertisement and rest 37 are for data transfer. Each channel is of 2 Mhz width.

There are two phases of communication; First is advertising.

Advertising phase is required for device discovery (broadcast and connection establishment) so that devices can contact each other.

There are 3 channels available for advertising phase.

Next phase is the data transfer (after connection establishment) that uses the rest of the 37 data channels.

## Channel Hopping

- Hop along 37 data channels
- One data packet per channel
- Next channel = current channel + hop increment (mod 37)
- Time between hops: hop interval, it is the duration when both communication parties stays in one channel. It is equal to one Round Trip Time + channel switch latency.

3 → 10 → 17 → 24 → 31 → 1 → 8 → 15 → ... (hop increment = 7)



One main observation is that you can hop along 37 channels and transmit one data packet per channel in a round robin fashion.

If you go beyond 37, you wrap around and select a channel starting from lowest number (1 if the channel increment is 1).

The next channel is the current channel + hop increment modulo of 37

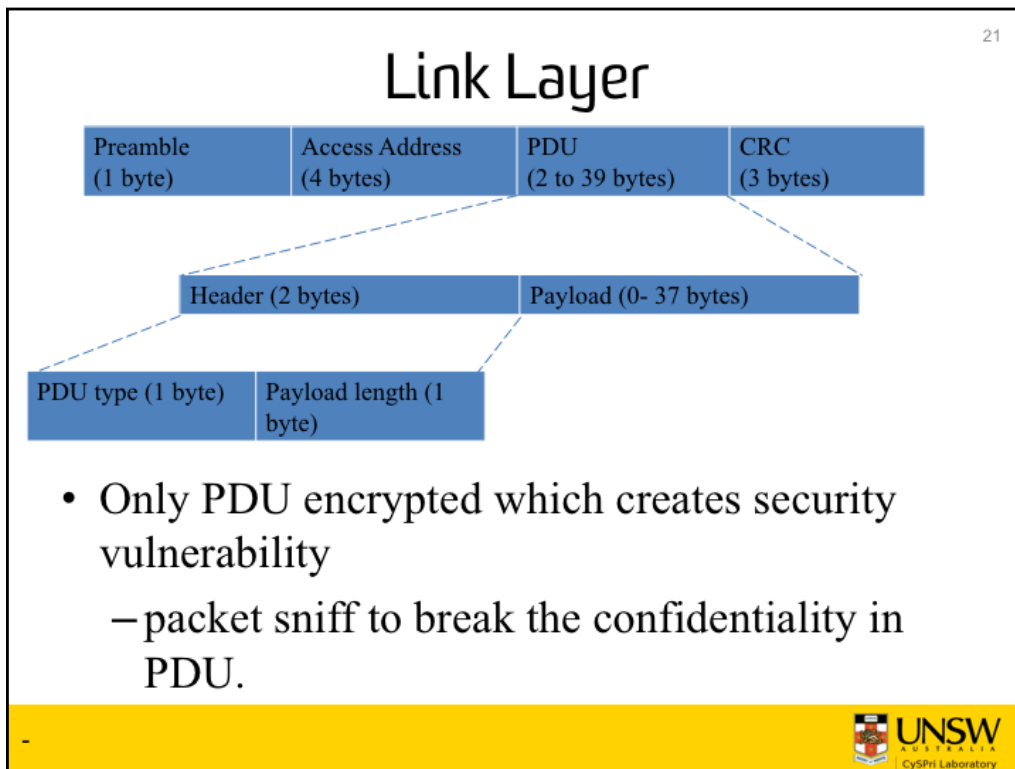
Hop increment is the number of channels skipped while selecting the next channel. You use mod 37 in order to always select a channel value that is  $\leq 37$ .

Hop interval is the time between each hop or the time both devices spend in one channel. = RTT + channel switching latency. Not always the same otherwise it would be very easy to guess.

Communicating parties negotiate Hop increment and Hop interval in connection establishment phase.

Example with hop increment of 7. After you hit 31, add 7 = 38 % 37 is 1. and so on.

37 is a prime. It is important.



Preamble: 8-bit sequence (01010101), it is for receiving radio to synchronize the transmission.

Access Address: it is a 32-bit address for recipient to pickup the intended Bluetooth packets.

PDU = Protocol Data Unit

CRC = Cyclic Redundancy Check

In each Bluetooth packet, you have 4 parts. 1 byte Preamble, 4 bytes Access address, (2-39 bytes) PDU and 3 bytes CRC.

CRC provides the integrity check whether the packet has experienced any bit corruption.

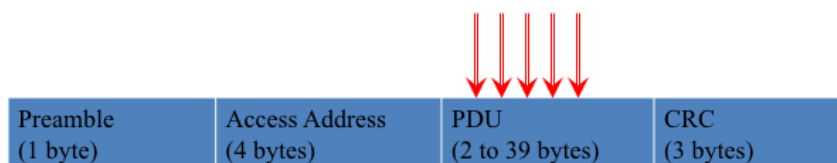
In actual PDU, there is a 2 byte header and up to 37 bytes of payload.

Note that only the PDU gets encrypted using the Session key (SK) that we discussed earlier, rest three (Preamble, Access address and CRC) are sent in plaintext.

Packet sniffing can be used to break the confidentiality.

# Encryption and MACs

- Encrypts and MACs PDU section
- AES-CCM algorithm
- AES-CCM is secure but the key exchange protocol is weak!



AEA-CCM algorithm is used for encryption that is secure but relies on the Temporary Key which is very easy to crack.

## Packet Sniff Process

- Configuration
  - Set modulation parameters to match BTLE (e.g., set to the same frequency, 2.4GHz)
  - Tune to proper channel – it needs to know the channel hopping pattern.
    - Hop Increment
    - Hop interval
    - Both can be sniffed from connection packet or recovery in promiscuous mode

-



First configuration of the packet sniffing process at the attacker, we set the modulation parameters to match BTLE (2.4 GHz). Secondly, it need to tune to the proper channel.

Basically need to monitor 1-2 channels not all of them.

What the attacker needs to know?

Require channel hopping pattern: Hop increment and Hop duration, how long the parties would remain on a particular channel and which channel would be used as the next channel.

Two ways to get this pattern.

1. Sniff the connection packets in the advertisement phase. This is the easiest way. You can sniff the negotiating devices when they are setting up the scheduling for channel hopping to agree on these parameters so that they know how to hop. Have to listen on at most 3 advertisement channels.

Here we assume that the attacker has missed the connection advertising phase and data phase is in progress.

2. In data transfer stage, we can use pattern detection in promiscuous mode and use some mathematics to recover the hop increment and hop duration

# What Information do we need?

Preamble (1 byte)	Access Address (4 bytes)	PDU (2 to 39 bytes)	CRC (3 bytes)
----------------------	-----------------------------	------------------------	------------------

- Access Address (AA)
  - Advertising: Fixed 0x8E89BED6
  - Connection: Actual device address
- Channel Information:
  - Hop interval
  - Hop increment

Where to get this info: **Connection packet!**

- easy if you get the starting packets

What information do we need?

1. Need Access Address to know if this communication is of our interest.
2. The other is the PDU that contains the payload having channel information (hop interval and increment) in the advertising phase.

Channel information is required so that the attacker know how the communicating parties would send the data using the channel hopping mechanism.

In the advertising phase all devices uses a fixed 0x8E89BED6 as Access address while in data phase it has the actual device address when they are exchanging data.

So it is very easy to identify packets exchanged in advertising phase (on one of three advertisement channels)

Channel information is contained in the connection packets exchanged during the advertising phase.

Devices negotiate to come in agreement of how much is hop increment and what is the hop interval.

Wait for the connection packet to arrive and retrieve the PDU. The connection packet contains the channel hop increment, hop interval in the payload.

Easy if you can capture the starting packets



## Promiscuous mode

- What if I missed the connection packets?
  - Capture a number of data packets.
  - Perform the pattern search (promiscuous mode) to recover access addresses, hop interval and hop increment values (**easily done!**)
- Crack the session key to decrypt PDU

-

The advertising phase is very short so most of the time the attacker will miss this phase (and miss the connection packets!).

Well what to do now?

In this case, when the communicating devices are exchanging data packets, attacker can capture a number of data packets to perform pattern search to recover the channel hopping information.

# Recovery of Access Address

Preamble (1 byte)	Access Address (4 bytes)	PDU (2 to 39 bytes)	CRC (3 bytes)
----------------------	-----------------------------	------------------------	------------------

What we know: Preamble (01010101)

What we have: Sea of bits

What we want: Access Address

10001110111101010101 → likely preamble!

10011100000100011001.. → part of AA

100011001...100011101 → 32 bit complete of AA! After that, PDU!

I will first talk about how to recover the Access Address (AA) so that the Attacker knows whether this packet is of interest.

When capturing, attacker gets sea of bits. However, one thing that the attacker would look for is the Preamble sent in the beginning of a packet.

It is always of a fixed pattern.

What the attacker would do in the huge number of bits (captured using Wireshark e.g.,) is to match the preamble pattern of specific 8 bits.

If you get to the preamble, next bits following it would be part of AA.

According to the format of the packet, after 1 byte of preamble is the 4 bytes of AA and remember AA is not encrypted and send in clear.

So the attacker can simply read the next 32 bits to read the AA.

Data followed 32 bits of AA is likely to be PDU.

It is easy to get the AA.

## Recovery of Access Address (Cont.)

- A preamble is “01010101” but “01010101” is not always a preamble.
- CRC is here to help (for attacker)!
- Attacker could use CRC (after PDU) to verify the access address and PDU.
  - If CRC passes, the access address is correct. Otherwise, the “01010101” is false positive for preamble.

-



But here comes a question.

The preamble pattern is fixed but every such pattern is not a preamble. This pattern can appear anywhere in the packet. How would the attacker make sure that the bit pattern he has found is indeed a preamble? Could be a false positive, if the pattern is not a preamble.

Attacker can use the CRC to verify that the pattern is indeed a preamble. CRC is to ensure that previous part of packets are not corrupted in transmission. Recalculate the CRC based on the position of pattern matched preamble and if it matches with the captured CRC, the pattern is an actual preamble and not a false positive.

The PDU length is variable, 2-39 bytes? How to know where the CRC starts?

In Bluetooth, two connected devices need to exchange packets in each hop interval even though they do not have anything to transmit.

Those are empty packets. The PDUs of these empty packets are fixed to 2 bytes (header only) and the format and content is always the same.

Attacker can use those empty packets to infer the access addresses.

## Recovery of Hop Interval

- Observation: 37 is a prime
- Sit on one data channel and wait for two consecutive packets. Measure the time difference.

$$\Delta t / 37 = \text{hop interval}$$

After the attacker is able to retrieve the Access Address, he is able to identify the devices.

If it want to monitor data for any particular device, it can then launch the second phase of the attack.

First of all it want to discover the hop interval that is how long both devices would remain on a channel.

One observation here is that number of channels available in BT Smart are 37 and 37 is a prime number.

What the attacker would do is to sit on any particular data channel and get one data packet. Wait till it get the next data packet on the same channel and then measure the time difference between these two packets. This time difference divided by 37 is the hop interval. Because 37 is a prime number, in order to get back to the same channel, devices would have to hop 37 times. We thus measure time difference between two packets received on the same channel and divide it by 37 to get the hop interval.

Each channel would be used exactly once before the cycle comes back to the same channel after 37 hops because 37 is a prime.

Example Hop increment of 1 would come back after 37, 2 would also come back after 37 hop intervals and so on.

## Recovery of Hop Increment

- Start on data channel 0, jump to data channel 1 when a packet arrives.
- We know hop interval, we can calculate how many channels have been hopped between channel 0 and 1.
  - $\Delta t / \text{hop interval} = \text{channel hops}$

-



For hop increment, it is a little bit tricky. Hop increment is how many channels have it skipped/hopped in one hop?

In the previous example, we have 7 channel hopping resulting in hop increment of 7.

For this, we monitor two channels to get the hop increment. Start from channel 0 and as soon as packet is received on channel 0, hop to next channel 1.

Measure the time difference when another data packet is received on channel 1. We know the hop interval (previous slide), time difference divided by hop interval would give as the count of how many channels have been hopped.

After this we have some mathematical computations to find the hop increment.

## Calculate Hop Increment

$$\text{HopIncrement} * \text{channel hops} \equiv 1 \pmod{37}$$

$$\text{HopIncrement} \equiv \text{channel hops}^{-1} \pmod{37}$$

Apply Fermat's little theorem :  $a^{p-1} \equiv 1 \pmod{p}$ ,

$$\text{HopIncrement} \equiv \text{channel hops}^{37-2} \pmod{37}$$

-



The hop increment is what the attacker would like to know.

Hop increment multiplied by channels hops (measured in last slide) = 1 (one channel difference ( channel 0 and channel 1) between two data packets) (mod 37)

Move channel hops to the other side it would become  $\text{HopIncrement} = \text{channel hops}^{-1} \pmod{37}$

Fermat's little theorem is given a Prime P, if integer a is not divisible by P,  $a^{(P-1)} \equiv 1 \pmod{P}$ .

Therefore,  $\text{channel hops}^{(37-1)} \pmod{37} \equiv \text{HopIncrement} * \text{channel hops}$ .

$$\text{HopIncrement} \equiv \text{channel hops}^{37-2} \pmod{37}$$

## Sniff summary

- Connections packets
- Promiscuous mode: recovery of
  - Access Address
  - Hop Interval
  - Hop Increment

-

What the attacker has now is the Access Address using the preamble and CRC, Hop Interval by monitoring one channel and Hop increment by monitoring two channels

With Hop interval and hop increment known, we have already cracked the channel hopping mechanism as the attacker can follow the hopping pattern and it does not have to monitor all 37 data channels simultaneously. For cracking, we have only monitored maximum of two data channels, one at a time instead of 37.

# Custom Key exchange protocol

32

- Three pairing methods
  - Just Works™
  - 6-digit PIN
  - 00B
  - “None of these key pairing methods provide protection against a passive eavesdropper” – Bluetooth Core Spec



There still remains one thing un-cracked that is the PDU. This part is encrypted using the Session Key (SK) mechanism discussed in previous slides.

For key exchange in Bluetooth, there are currently three pairing methods JustWorks, 6-digit Pin and 00B. However, the Bluetooth core specifications states that none of these paring methods provide protection against a passive eavesdropper.



## Cracking Temporary Key

- **Temporary Key (TK)** = AES (**PIN**, AES(**PIN**, **rand** XOR **p1**) XOR **p2**) ( $E_2$  in slide 10)

**Green** – transmitted in plaintext

**Red** – wanted to know

PIN: integer between 0 and 999,999

JustWork™ is always 0!

-



Recall that the Temporary key TK is used for authentication. Here is the formula to generate the TK.

The Green one are transmitted in plaintext and known to the communicating parties and the attacker.

The Red one are only known by the communicating Bluetooth devices.

The temporary key is thus only known to the communicating devices.

The PIN is used to generate a TK and TK is used to generate the SK.

PIN values ranges from 0 to 999999 (1 million) for 6-digit PIN and the worst case in JustWorks it is always 0.

## Cracking the PIN

Total Time to crack:  
< 1 second

-



Time to crack the PIN is less than 1 sec. Even with PIN values from range 0 to 999999, with super computers we can brute force very easily.

## Subsequent Key crack

PIN → STK

STK → LTK

LTK → Session key!

- Every key is known.
- Attacker can learn about PDU
- Attacker can inject the packets in the networks with the session key!

-



With PIN you can know the Temporary key, with Temporary Key you can generate the Session Key. You already know the channel hopping. Infact, it is easier to crack PIN than Channel hopping as you need some mathematical computations in cracking channel hopping increment.

Terminologies of BTLP (Smart)

STK = Short Term Key

LTK= Long term Key

# Roadmap

- Introduction
- Features
- Security Issues
- Case Study – packet sniff
- **Conclusions**

## Conclusions


- Bluetooth has some security mechanisms
- Bluetooth is not secure. There exist loopholes and they are easy to exploit.
- Security in Bluetooth is yet to be improved.

Bluetooth have some security mechanisms but it is not secure at all.  
There exist loopholes everywhere and they are easy to exploit.  
So security in Bluetooth is yet to be improved  
Unfortunately not many effective methods are in use.

38

Thank you!

Any Questions?

UNSW  
AUSTRALIA  
Cy5PI Laboratory

Any Questions?

I hope I have not scared you off using the Bluetooth.

Q1: Why do we use CRC to provide data integrity as it would slow down the communication? Maybe some applications don't care for data integrity.

Performance of the protocol may suffer due to lack of CRC. Tradeoff performance vs security & reliability.

Q2: All the applications that are using Bluetooth are low –security based such as listening to music on headphones. Why need all these security mechanisms?

Password on a Bluetooth enabled keyboard can be captured if sending without confidentiality.