FFI-rapport 2015/01091

Bluetooth security and threats

Vinh Pham and Janne Hagen

FFI Forsvarets forskningsinstitutt

Bluetooth security and threats

Vinh Pham and Janne Hagen

Norwegian Defence Research Establishment (FFI)

17 November 2015

FFI-rapport 2015/01091

1242

P: ISBN 978-82-464-2616-7
E: ISBN 978-82-464-2617-4

Keywords

Bluetooth

Trådløs kommunikasjon

Sikkerhet

Kryptering

Approved by

Ronny Windvik	Project Manager
Anders Eggen	Director

English Summary

Since its birth in 1994, Bluetooth has gained increasingly higher popularity and acceptance. The technology has now become a de facto standard for short range wireless communication, and Bluetooth is applied in a diversity of devices including consumer electronics, health care, transportation and in industry. Bluetooth is also anticipated to be used in military applications, e.g. as a wireless link between a personal digital assistant (PDA) and a Harris long range radio.

The proliferation of the technology has naturally led to increased focus on the security as information is transmitted over the unsecure wireless medium, and hence is vulnerable for a variety of attacks including eavesdropping and denial-of-service (DoS) attacks. Due to the popularity, the security in Bluetooth plays a very important role, first of all in public society, and also, in a long term perspective, for the Armed Forces. This study gives an in-depth overview of the security architecture and mechanisms of Bluetooth, as well as its weaknesses and vulnerabilities.

Bluetooth has evolved over time in terms of capability and security. This is reflected through the different Bluetooth specification versions: Bluetooth Classic, also referred to as Basic Rate/ Enhanced Data Rate (BR/EDR), Alternate MAC/PHY (AMP), and Low Energy (LE). In parallel, the security in Bluetooth has gradually evolved in three phases, referred to as Legacy Security, Secure Simple Pairing (SSP), and Secure Connections. In Legacy Security, the link key is derived from the PIN code, which provides low entropy and hence vulnerability to exhaustive search attacks. In SSP, important changes have been made in the security architecture in order to improve the weaknesses seen in Legacy Security. Elliptic Curve Diffie-Hellman (ECDH) and association models have been introduced (among others), which provide enhanced security, increased user friendliness, and stronger protection against exhaustive attacks. The latest security architecture, Secure Connections, is based on SSP. The difference is that Secure Connections utilizes even stronger and more secure cryptographic algorithms compared to SSP. Additionally, it also supports integrity protection.

A number of security breaches and threats have been discovered over the years. In the most severe cases, an attacker may gain access to restricted data, or even worse, take complete control of the target Bluetooth device, e.g. a mobile phone. These security threats are due to either weaknesses in the security architecture (especially in earlier versions of the Bluetooth specification), side-effects of design features, improper implementations by the manufacturers or improper use by the user.

Most of the security threats and vulnerabilities discovered are probably obsolete since the introduction of SSP. However, even though the security has been improved, there are still weaknesses in the Bluetooth security architecture as documented in several publications. This is due to the fact that it is very difficult to design a security architecture that is secure and user friendly at the same time. Often, a compromise is made between security and usability. This is also the case for Bluetooth. From a user point of view, it is important to be aware of the potential risk and learn how to protect the Bluetooth device.

Sammendrag

Siden unnfangelsen i 1994 har blåtann gradvis blitt mer populær og utbredt, og teknologien er nå en de facto standard for trådløs kommunikasjon over kort rekkevidde. Blåtann anvendes i et mangfold av innretninger innen områder som forbrukerelektronikk, helse, transport og industri. Også i militær sammenheng ser en for seg mange ulike anvendelser. Et eksempel er bruken av blåtann som trådløs forbindelse mellom en personlig digital assistent (PDA) og en Harris radio.

Utbredelsen av teknologien medfører større fokus på sikkerheten ettersom informasjon overføres via et usikret trådløst medium som dermed er potensielt utsatt for ulike former for angrep, for eksempel avlytting og tjenestenektangrep. Populariteten til blåtann gjør at sikkerheten spiller en svært viktig rolle, først og fremst for det sivile samfunnet, men på sikt også for Forsvaret. Denne studien tar for seg sikkerhetsarkitekturen og sikkerhetsmekanismer i blåtann samt dens svakheter og sårbarheter.

Blåtann har utviklet seg over tid både med tanke på kapabilitet og sikkerhet. Dette gjenspeiles i de ulike blåtannversjonene som finnes: Basic Rate/ Enhanced Data Rate (BR/EDR), Alternate MAC/PHY (AMP) og Low Energy (LE). Parallelt har sikkerheten i blåtann utviklet seg gradvis i tre faser, omtalt i spesifikasjonen som Legacy Security, Secure Simple Pairing (SSP) og Secure Connections. I Legacy Security blir linknøkkelen utledet direkte fra pinkoden, hvilket resulterer i lav entropi og sårbarhet for angrep basert på uttømmende søk. I SSP er det gjort viktige endringer i sikkerhetsarkitekturen for å forbedre svakhetene i Legacy Security. Det er blant annet innført Elliptic Curve Diffie-Hellman (ECDH) og assosiasjonsmodeller som gir bedre sikkerhet, økt brukervennlighet og bedre beskyttelse mot angrep basert på uttømmende søk. Secure Connections er i bunn basert på SSP. Forskjellen er at Secure Connections bruker enda sterkere og sikrere kryptografiske algoritmer og i tillegg har støtte for integritetsbeskyttelse.

Sikkerhetsbrudd og -trusler har blitt oppdaget. I de mest alvorlige tilfellene kan en angriper aksessere begrenset data eller ta fullstendig kontroll over blåtanninnretningen, eksempelvis en mobiltelefon. Disse sikkerhetsbruddene og truslene har oppstått på grunn av svakheter i sikkerhetsarkitekturen (spesielt i tidligere versjoner), bieffekter av design, feilimplementasjon fra produsent og brukerfeil.

De fleste avdekkede sikkerhetssvakhetene i blåtann er trolig foreldet etter introduksjonen av SSP, men til tross for vesentlige forbedringer er det fortsatt sårbarheter i sikkerhetsarkitekturen, noe som er dokumentert i flere publikasjoner. Dette skyldes i hovedsak at det er svært vanskelig å designe en sikkerhetsarkitektur som også er brukervennlig. I praksis blir det ofte et kompromiss mellom sikkerhet og brukervennlighet, noe som også er tilfellet for blåtannteknologien. Sett fra et brukerperspektiv er det derfor viktig å være klar over risikoen og lære hvordan en kan beskytte enheter som kommuniserer via blåtann.

Contents

1 Introduction

Bluetooth is a technology that enables short range ad hoc wireless communication. The technology enables users to establish personal network, and has been designed to operate in noisy radio frequency environments using frequency-hopping scheme to make the link robust. It is used in a range of applications, for instance in consumer electronics, health applications, industrial control systems such as Supervisory Control and Data Acquisition (SCADA), business and even security applications.

The Bluetooth standard is managed by the Bluetooth SIG Inc., which has 24,000 member companies worldwide that jointly drive the Bluetooth developments. Among these are large international hardware and software producers, including Ericsson, Intel, Nokia, IBM, Toshiba etc. The trend is that Bluetooth will be embedded in even more devices in the future, especially with the advents of Internet of Things (IoT) and Bringing Your Own Device (BYOD).

The proliferation of Bluetooth has called for greater focus on the security mechanisms of the standard. The motivation of this report is therefore to provide an overview of security features provided by the Bluetooth technology. Furthermore, this report also provides an overview of vulnerabilities related to Bluetooth security, as well as its evolvement throughout the years to mitigate potential threats. Both functionality and security has improved as time passed, meaning that newer versions are in general better protected than the older ones. However, as discussed later in this report, there are still weaknesses even in the newer versions of the Bluetooth standard.

How vulnerable devices are depends thus on the embedded Bluetooth version. Because of budget constraints and because it takes time to renew technology in public administration, businesses and in private homes, it can be expected that some users will carry vulnerable Bluetooth devices also in the future. The backward compatibility of Bluetooth makes it just as secure as the weakest link of a Bluetooth connection.

The scope of this report is mainly limited to Bluetooth Classic, also referred as Basic Rate/Enhanced Data Rate (BR/EDR). We omit the security in Bluetooth Alternate MAC/PHY (AMP, version 3.0) as well as Bluetooth Low Energy (LE). The reason for this is due to the plural number of Bluetooth versions that exist, and not the least the length of the Bluetooth Specification of almost 3000 pages, the limited time frame and the size of this report. Bluetooth AMP is anticipated to not be very widespread. The same may be said about Bluetooth LE at present. However, it is expected that this technology will gain rapid growth in the near future in conjunction with the proliferation of Internet of Things (IoT). Bluetooth Classic is on the other hand, based on a market report[1] from Bluetooth SIG, the dominating technology and will continue to be the dominating technology in close future.

[1] https://www.bluetooth.org/en-us/marketing/market-research

The report is divided in 8 chapters, where Chapter 2 provides an overview of security features provided by the Bluetooth standard. Chapter 3, 4 and 5 provides an in-depth description of the three main security architectures, i.e. Legacy Security used in version 2.0 and earlier versions, Secure Simple Pairing (SSP) applied in version 2.1 and later versions, and Secure Connections introduced in version 4.1. In Chapter 6, an overview of known vulnerabilities and threats is given, while Chapter 7 provides measures to mitigate potential threats. Finally, a conclusion is given in Chapter 8.

For an introduction to the Bluetooth technology and how it works, it is recommended to read FFI-report 2015/00293.

2 Security Overview

2.1 Security Threats and Vulnerabilities

Bluetooth is exposed for the same kind of threats as other wireless communication technologies. Haataja (2009) divides security threats in distributed networks into three categories:

- Confidentiality or disclosure threats, where information leaks to an unauthorized eavesdropper
- Integrity threats, where information is altered and mislead the authorized user
- DoS-threats, which blocks the access to a service and hence makes it unavailable for an authorized user

Among these threats, the DoS threat is considered to be more annoying than damaging, but it can be part of a major attack campaign and paralyzing the system of the victim.

As to be discussed in the following sections, the Bluetooth Specification addresses some of these threats, but not all. First of all, all versions of Bluetooth provide protection mechanism against confidentiality threats, as this is one of the most crucial feature in any security system. Bluetooth utilizes Frequency Hopping Spread Spectrum (FHSS), which makes it more difficult and challenging for an adversary to perform eavesdropping. In addition, this radio scheme provides enhanced protection against interference and to a certain extent DoS-threats. In terms of integrity threats, there was no such protection until Bluetooth version 4.1, when Secure Connections was introduced. This specification provides support for integrity protection with cryptographic strength. In terms of DoS threats, there is no mechanism in the specification that protects against this kind of threat. However, the risk for DoS is probably small since the Bluetooth technology, by nature is a short range wireless system. This implies that unless the attacker has specialized equipment, common Bluetooth devices can only perform such an attack in close proximity, and may thus risk to be detected.

When studying technical systems, it should be emphasized that human still are the weakest link, just as the security expert Bruce Schneier stated (2004). Tan and Masagca (2011) studied undergraduate students' use of Bluetooth and their security attitudes and behaviour. Of the 400 student surveyed, 84 % were considered regular users of Bluetooth applications. File and data exchange, and establishing connection with wireless accessories, were the main applications. They also found that a higher portion of business students regarded Bluetooth as secure (71 %) compared with engineering students (46 %). Despite that, more than half of the students were aware of the security threats, only 33% protected themselves. This reveals a gap between security attitudes and security behavior which is consistent with research on human security behavior (Hagen, 2009). As long as the technical security mechanisms do not close this gap, there will always be a presence of security weaknesses that are susceptible for exploitation, for instance through social engineering attacks.

2.2 Security Features Overview

Bluetooth security includes five distinct security mechanisms:

1. *Pairing*: the process of creating a shared and secret link key between two connecting Bluetooth devices.
2. *Bonding*: the act of storing the link key created during pairing for later use in future connection establishments. Bonding thus establishes a semi-permanent trusted relationship between two Bluetooth devices, and simplifies as well as makes future connection establishments between the bonded devices to occur faster.
3. *Device authentication*: the act of verifying that two devices have the same link key. Bluetooth provides only authentication at device-level. User-level authentication may be implemented at the application layer by third party software.
4. *Encryption*: provides message confidentiality through transforming clear text to cipher text.
5. *Message Integrity*: protects against modifications and forgeries of message exchanged between two Bluetooth devices.

The first two points are concerned about establishing a shared secret key in which a mutual trusted relationship between two connecting Bluetooth devices is formed. On the other hand, the bottom three points are the main security mechanisms provided by Bluetooth. These mechanisms rely on that the key establishment process in the first two points is properly handled, i.e. secrecy is preserved, in order to not be compromised. All Bluetooth versions support the first four points. On the contrary, the last point is only supported in version 4.1 (Secure Connections).

2.3 Security Modes

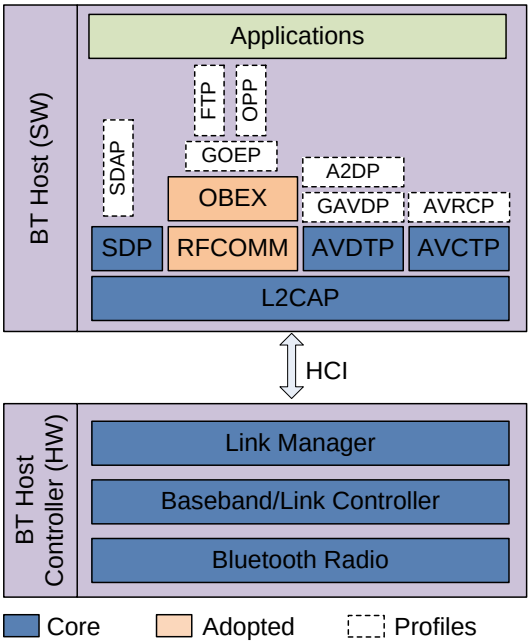

Figure 2.1 The Bluetooth protocol stack.

Cumulatively, the various versions of Bluetooth specifications define four security modes. Security mode 1-3 are referred to as legacy security modes, and apply to Bluetooth devices that implement version 2.0+EDR or earlier. These modes utilize *legacy security* (described in Chapter 3), i.e. the classic or old way to perform security feature. Security mode 4 applies to devices that support *Secure Simple Pairing* (SSP), implemented in version 2.1+EDR or later. SSP is a better security architecture compared with older versions.

Each version of Bluetooth supports some of the security modes, but not all four modes. A Bluetooth device may support two security modes simultaneously: security mode 2 for backwards compatibility with remote devices that do not support SSP, and security mode 4 for devices that support SSP.

Security Mode 1 is a non-secure operational mode. None of the supported security mechanisms like authentication and encryption is utilized in this mode. A Bluetooth device in this mode will therefore allow any other Bluetooth devices to connect. Security Mode 1 is only supported in Bluetooth version 2.0+EDR and earlier.

Security Mode 2 is a service-level security mode which implies that security procedures are initiated after link establishment at the Link Manager layer but before logical channel establishment at the Logical Link Control and Adaption (L2CAP) layer (see Figure 2.1). L2CAP resides in the data link layer and provides connection-oriented and connectionless data services to upper layers. In this mode, a security manager maintains policies for access control to services and devices. Various security policies and trust levels can be applied on applications with different security requirements to provide access to specific services while restricting others.

These security policies provide in other words the notion of authorization or access control to services offered by a Bluetooth device. The authentication and encryption mechanisms used for security mode 2 are implemented at the Link Manager layer. Security Mode 2 is supported by all Bluetooth versions, but from version 2.1 and later, this mode is only supported for backward compatibility.

Security Mode 3 is a link level security mode in which security procedures are initiated before the physical link is fully established. A Bluetooth device operating in security mode 3 provides authentication and encryption for all connections to and from the device. The authentication and encryption features are based on a secret shared link key that is generated during pairing. Security mode 3 is only supported in version 2.0 + EDR and earlier devices.

Security Mode 4 is a service-level enforced security mode and was introduced in Bluetooth v.2.1 + EDR. Security Mode 4 is similar to Security Mode 2 with the exception that it uses the new pairing scheme called Secure Simple Pairing (SSP). SSP simplifies and improves the pairing process by using public key cryptography and *association models* (as discussed in later subsections). Authentication and encryption algorithms are identical to the algorithms in security mode 2. Only devices with Bluetooth v.2.1 + EDR or later can use this Security Mode.

2.4 Connection Establishment Sequence

Establishing a connection between two devices involves a sequence of procedures that must be performed. Figure 2.2 shows a simplified flow diagram of procedures performed during a life-cycle of a connection, from establishment to disconnection. The intention of the diagram is to give the reader a better understanding of the context of the security procedures, i.e. when are these security procedures actually performed during a connection establishment.

Initially, an inquiry procedure is performed to search for nearby devices. Once a peer device is discovered, a paging procedure is performed to establish a connection between the two devices. The device which initiates the inquiry and paging procedure is referred to as the master while the responding device is the slave. After the paging is successfully executed, the slave is synchronized in terms of frequency hopping pattern and clock timing with the master.

The next steps in the connection establishment sequence are security procedures and include, pairing, authentication, generation of encryption key, data encryption. Each of these security procedures is optional. Which security procedure to execute is dependent on the applied security mode. This provides some flexibility for an application to decide the appropriate security measures to be appended.

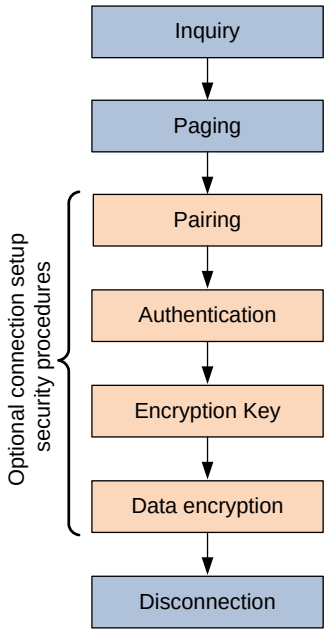

Figure 2.2 Simplified connection establishment procedure

The above pattern of security procedures is very characteristic in Bluetooth, and the very same pattern will appear again in several contexts in subsequent sections. The inquiry and paging procedure is described in an earlier Bluetooth introduction report (Hagen and Pham 2015) and is hence omitted in this report. Instead, much of this report is devoted to describe the security procedures and mechanisms in detail, in subsequent sections.

Finally, when the connection is not needed anymore, the disconnection procedure is performed to release the connection.

2.5 Security Architecture Evolution

The Bluetooth technology and its security architecture have evolved over the years. Haataja (2009) gives an overview of how data is encrypted and secured up to Bluetooth version 3.0. In general, newer versions have more security features and are more secure than the older versions. Older versions have security weaknesses that are well documented in the literature. These weaknesses are removed or improved in never versions.

Below follows a summary of the most important changes over time:

- The security mechanisms in Bluetooth version 2.0 and earlier is often referred to as *legacy security* or *legacy pairing*.
- In Bluetooth version 2.1 + Enhanced Data Rate (EDR), a new security architecture is introduced called *Secure Simple Pairing* (SSP). SSP utilizes Federal Information Processing Standards (FIPS) approved algorithms like SHA-256, HMAC-SHA-256 and P-192 elliptic curve.

- Version 3.0 + HS of the Core Specification added support for Alternate MAC/PHY (AMP), which is an extension that allows for the utilization of an alternate radio technology in parallel, e.g. Wifi, in order to increase data rate and transfer bulk data. AMP security is not covered in this report as this version is anticipated to not be widely used.
- Version 4.0 of the Core Specification added entire security model for Low Energy (LE). A major difference between LE and BR/EDR, is that security mechanisms in LE are moved up in the protocol stack compared to BR/EDR. While the security mechanisms in BR/EDR are implemented in the Link Manager Protocol on the Controller, on the other hand, LE implements most of its security mechanisms in the Security Manager on the Host. The motivation of this change is to keep the cost of LE controllers low and secondly to provide more flexibility to the host. The security functionality for BR/EDR and AMP remained unchanged.
- Version 4.1 added the Secure Connections feature, which is actually based on SSP. In Secure Connections the default cryptographic algorithms are replaced with even stronger algorithms, approved by FIPS. These algorithms include P-256 elliptic curve for public key exchange, HMAC-SHA-256 and AES-CTR for authentication, and AES-CCM for message confidentiality and integrity.

As can be seen from the evolution above, the Bluetooth Specification includes several types of controllers (or radio hardware) referred to as BR/EDR, AMP and LE. Each type of controller has its own protocol and security architecture. A complete security overview for all these controllers is beyond the scope of this report. Instead, this report focuses on the security in Bluetooth BR/EDR as this is the most widespread Bluetooth controllers today.

To summarize, the security mechanisms used in Basic Rate/Enhanced Data Rate (BR/EDR) have evolved over the course of multiple Core Specifications in three phases: legacy security, SSP, and Secure Connections. The difference between these security schemes with respect to encryption, authentication and key generation algorithms is summarized in Table 2.1. In the next chapters a detailed description of each of these security schemes are provided.

Table 2.1 Comparison of Security Schemes in Bluetooth

Security Mechanism	Legacy	Secure Simple Pairing	Secure Connections
Encryption	E0	E0	AES-CCM
Authentication	SAFER+	SAFER+	HMAC-SHA-256
Key Generation	SAFER+	P-192 ECDH HMAC-SHA-256	P-256 ECDH HMAC-SHA-256

3 Legacy Security

Legacy security refers to Bluetooth version 2.0 or earlier, and uses the legacy pairing procedure (also called LMP pairing since all security procedures are executed using the Link Manager Protocol at the Link Manager layer) to establish a secret link key between to Bluetooth devices. The pairing procedure is based on the usage of a PIN code as input to generate the secret link key. The whole security chain of events, referred to as the initialization procedure, is illustrated in *Figure 3.1* and consists of three main phases:

1. Link key establishment (pairing)
2. Authentication
3. Encryption.

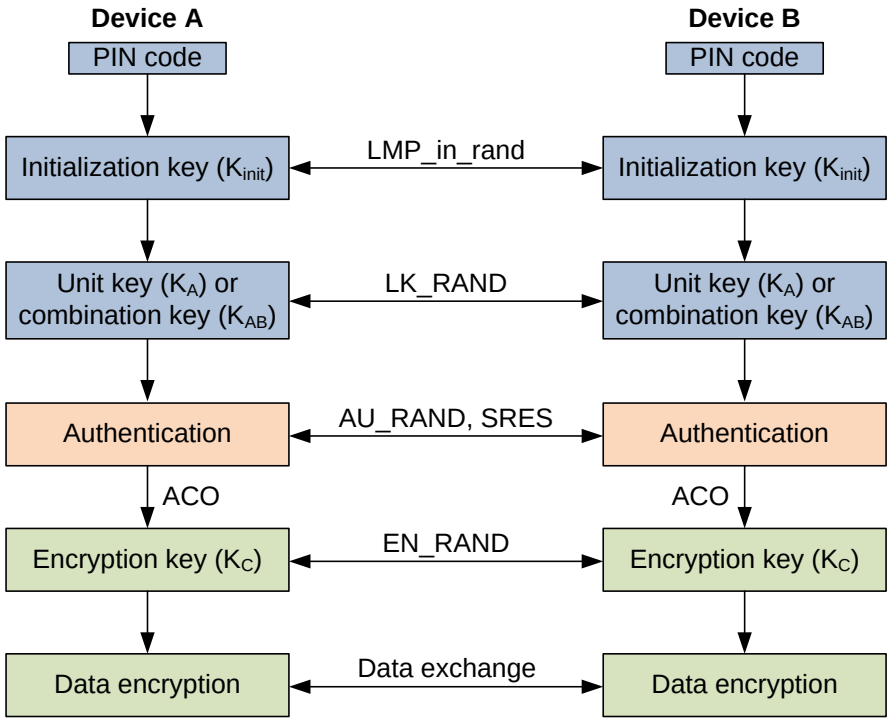

Figure 3.1 Initialization procedure

3.1 Link Key Establishment

In order to accommodate different types of applications, four types of link keys have been defined:

- Initialization key K_{init}
- Unit key K_A,
- Combination key K_{AB}
- Master key K_{master}

The following subsections describe the purpose of these keys as well as how they are generated.

3.1.1 Generation of Initialization Key

Initially, when device A and B meet for the first time, an initialization key K_{init} is generated and used as a temporary link key. This temporary key is then used to protect initialization parameters subsequently exchanged between A and B. The generation of K_{init} is based on the following input parameters:

- a PIN of L octets long
- a Bluetooth device address BD_ADDR
- a 128-bit random number IN_RAND

The random value IN_RAND is generated either by A or B and sent over the air in clear text to the other party. The party who generates and sends the random value IN_RAND is called the *initiator*, while the receiving party is called the *responder*. Upon receiving IN_RAND, the responder replies with an LMP_accepted message and both initiator and responder can now calculate K_{init} using the E_{22} key generation function:

$$K_{init} = E_{22}(PIN', IN\_RAND, L') \qquad (3.1)$$

Note that in the above equation, the PIN code and its length L (between 1-16 octets) are modified into PIN' and L' before applying to the key generation function E_{22}. If the PIN code is less than 16 octets long, then part of, or the entire BD_ADDR is appended to the PIN code, such that the resulting PIN' is either 16 octets long or equal to L plus the length of BD_ADDR, which is 6 octets. For example, if a four octets PIN-code is used, i.e. L = 4, then the resulting PIN' = PIN ∪ BD_ADDR, and L'=4 + 6 = 10.

The responder's BD_ADDR is used if both devices have variable PIN. On the other hand if the responder has a fixed PIN, while initiator has a variable PIN, the responder must generate a new IN_RAND value and send it to the initiator, and the initiator's BD_ADDR is used instead. In the case where both have fixed PIN, then they cannot be paired. This procedure ensures that K_{init} depends on the identity of the device with a variable PIN.

3.1.2 Generation of Unit Key

The unit key K_A is generated by one device, typically the device with limited storage memory. Suppose device A is the one who generates K_A. Device A will then use its own address BD_ADDR_A, and a generated 128-bit random value $RAND_A$ as input to the key generation function E_{21}, as shown in Equation (3.2). The output is the unit key K_A.

$$K_A = E_{21}(RAND_A, BD\_ADDR_A) \qquad (3.2)$$

Next, the unit key is encrypted and sent to device B. The encryption of K_A is performed with the initialization key K_{init}, i.e. $K_A \oplus K_{init}$. Upon reception, device B decrypts the message using the

same K_{init} , i.e. $(K_A \oplus K_{init}) \oplus K_{init} = K_A$. Finally, A's unit key K_A is stored and used as the link key between A and B. Note that using the unit key as link key is deprecated, since it's implicitly insecure[2] and the use of it is not recommended. However, it is still supported due to backward compatibility. The reason for deprecation is due to the fact that the unit key is usually generated only once at installation of the Bluetooth device, and is rarely changed. Furthermore, a Bluetooth device that uses the unit key as link key, has only one key to use for all of its connections. This means that the same key is shared with all other trusted Bluetooth devices. This also implies that a trusted device possessing that key can eavesdrop on the traffic between any pair of Bluetooth devices using that key, and it is also possible to impersonate a target device by just duplicating its BD_ADDR. Another danger is once the unit key is shared to a single peer device, then the owner of this key does no longer have full control of the key, as it cannot prevent the peer device from sharing the key to another third part. Due to these reasons, the use of unit key should be avoided if possible because of its insecure nature.

3.1.3 Generation of Combination Key

A combination key is, on the other hand, derived using information from both devices. This implies that both devices actively participate in the key generation process. The procedure is as shown in Figure 3.2. Initially, device A and B generate each a random number $RAND_A$ and $RAND_B$, respectively. Each device then generates a unit key, K_A and K_B, using Equation (3.2).

Next, device A encrypts its random number and sends it to B, i.e. $C_A = RAND_A \oplus K_{init}$. The same operation is also performed in device B, i.e. B sends $C_B = RAND_B \oplus K_{init}$ to A. Upon reception of the encrypted random number, each device now decrypts the number, i.e. $RAND_A = C_A \oplus K_{init}$ and $RAND_B = C_B \oplus K_{init}$, and calculates the other party's unit key.

Finally, when both unit keys K_A and K_B are available, the combination key can be easily generated by XOR-ing the two unit keys, i.e. $K_{AB} = K_{BA} = K_A \oplus K_B = K_B \oplus K_A$. The whole operation of generating the combination key is expressed in Equation (3.3). The combination key is stored in device A as K_{AB}, and in device B as K_{BA}.

$$\begin{aligned} K_{AB} &= E_{21}(RAND_A, BD\_ADDR_A) \oplus E_{21}(RAND_B, BD\_ADDR_B) \\ &= K_A \oplus K_B \end{aligned} \qquad (3.3)$$

When both devices have derived the combination key, the initialization key K_{init} is no longer needed and can thus be deleted. Using the combination key as link key is much more secure compared to the unit key, as each pair of Bluetooth devices has its own unique link key.

[2] Bluetooth Specification Version 4.1, Vol. 2, page1294.

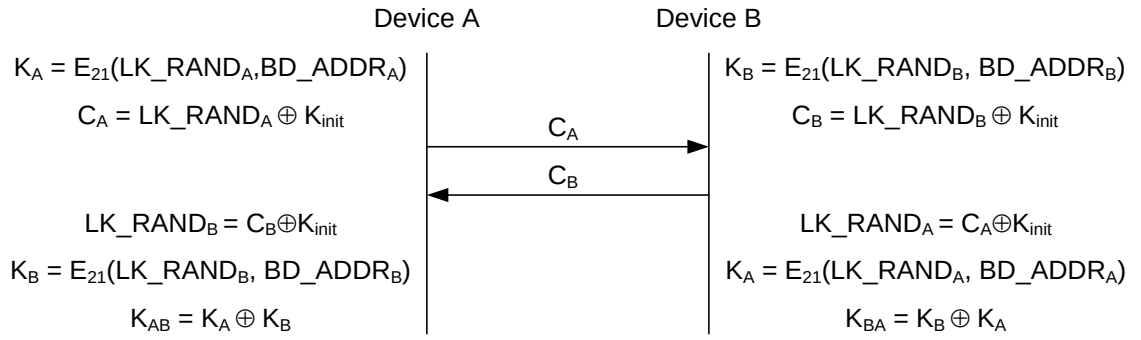

Figure 3.2 Procedure for generating the combination key

3.1.4 Generation of Master Key

When it is desirable for a master to encrypt broadcast traffic, the master can instruct the slaves to switch to a new temporary and common link key called the master key $K_{master.}$ The reason for this is twofold:

- When encrypting broadcast traffic or point-to-multipoint traffic, one may in theory use individual encryption keys for each link (as in the case of unicast). For example if a master A wants to broadcast some encrypted data to the slave devices, B and C. A can first send the data encrypted with the encryption key EK_{AB} to B, and then encrypt the same data again with the key EK_{AC} before sending the message to C. In total this equals to 2 transmissions. If there are more slaves in the piconet, the number of transmissions for each broadcast packet equals to the number of slaves, i.e. up to 7 slaves. Apparently, this approach is not very efficient and will result in waste of bandwidth and capacity loss for the piconet. Instead, using a shared encryption key (derived from the master key), it is sufficient to transmit a broadcast message once independently of the number of recipients.
- A slave might not be capable of switching between two or more encryption keys in real time. This implies that the master cannot use different encryption keys for broadcast and unicast messages. Consequently, if broadcast traffic is encrypted, then unicast traffic is also encrypted, and both are encrypted with the same encryption key.

The procedure for generating the master key is illustrated in Figure 3.3. Initially the master creates a new master link key from two 128-bit random numbers, RAND1 and RAND2. The third argument, i.e. value 16, is the output length in octets (16*8 = 128 bits). The output of the E_{22} function is the master link key K_{master} of 128-bit length.

Next, a third random number RAND3 is generated and sent to the slave(s). Upon receiving RAND3, the slave computes a temporary overlay key OVL, using the current link key $LK_{current}$, RAND3 and output length in octects (16 in the example) as input. This key is used to encrypt/decrypt the subsequent exchange of K_{master} from the master to the slave.

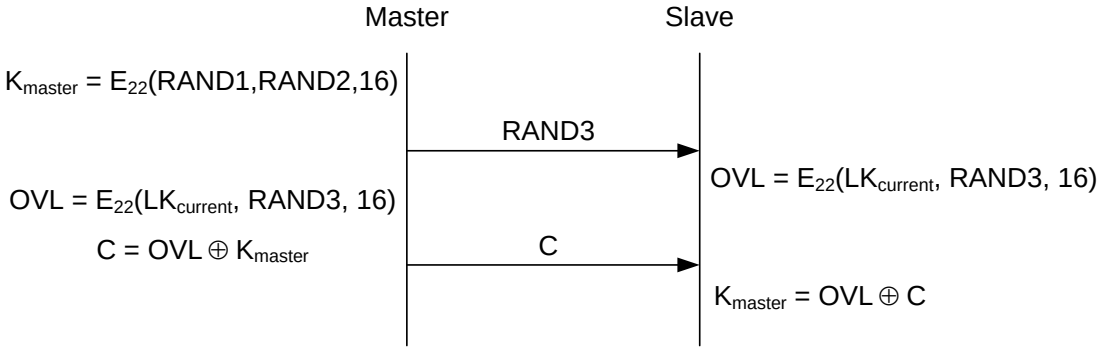

Figure 3.3 Procedure for generating the master key

3.1.5 Current Link Key

A node may, from time to time, change its current link key, i.e. the link key in use at the current moment. The current link key may either be the initialization key, the unit key, the combination key or the master key. Whether to use either of these keys depends on the scenario, device, application or type of traffic.

The initialization key is only used as link key in the initial phase when no unit key or combination key have been defined or when a link key has been lost.

A device with very limited memory capability will typically use the unit key as the current link key, since the device only has to remember its own unit key. Furthermore, devices that are installed in equipment that will be accessible by a large group of users may also use the unit key. However, as explained earlier, the use of the unit key should be avoided if possible due to its insecure nature.

Applications that require higher level of security should use the combination key, especially when encryption of unicast traffic is desired. As each link has its own encryption key, it is less susceptible for being eavesdropped like in the case with unit key. The drawback is that more memory is required, since each pair of device requires its own combination key. It is recommended to use the combination key as current link key when possible, since it is considered as more secure.

The master key is used to temporarily replace the current link key in order to facilitate encryption of broadcast traffic. The master key can be used to derive a common encryption key shared by multiple devices in the piconet. This allows for a more efficient way to encrypt broadcast traffic or point-to-multipoint traffic. Like in the case of unit key, each slave in possession of the master key can eavesdrop on all encrypted traffic, not only the traffic destined to itself.

3.1.6 Security Considerations

As shown above, with legacy pairing, the secrecy of the generated link key is dependent on the initialization key, which again is derived from the PIN. In this process, the PIN is the only source of entropy, which in many use cases, is typically four digits (resulting in only 10^4 possible PIN

combinations) either selected by the user or fixed for a given product. Therefore, if the pairing procedure and one authentication exchange is recorded one can run an exhaustive search to find the PIN in a very short amount of time on commonly available computing hardware, as demonstrated in (Shaked and Wool, 2005).

3.2 Authentication

Once a link key is created, the next step in the security chain of events is authentication, in which the purpose is to verify that both devices share the same secret link key. The authentication procedure for legacy pairing is called legacy authentication. It is based on a challenge-response scheme, where the authenticator called the *verifier* issues a challenge in which the device to be authenticated, called the *claimant*, must respond with a correct answer in order to successfully pass the test.

The authentication procedure is illustrated in Figure 3.4, and starts with the verifier generating a random number AU_RAND_A (which is the challenge) and sends it unencrypted to the claimant through an LMP_au_rand message. Next, both parties use the E_1 function to calculate a 32-bit value called Signed Response (SRES). The claimant then sends its SRES value unencrypted to the verifier through an LMP_sres message. The verifier then compares the received SRES with its own SRES', and if the values match, then the authentication is completed successfully.

The E_1 function is based on the SAFER+ block cipher (Massey et al., 1998) and is a computationally secure authentication algorithm[3]. As shown in *Figure* 3.4, the E_1 function takes as input the current link key LK, the random value AU_RAND_A, and the address BD_ADDR_B of the claimant. Since the link key is one of the inputs to the E_1 function, the generated SRES and SRES' can only be equal when both parties possess the same link key. Furthermore, in addition to the SRES value, the E_1 function also produces a 96-bit Authentication Ciphering Offset (ACO) value. This ACO value is used later on in the encryption key generation.

There is no requirement that the verifier is the master, but rather, it is up to the application to determine which party to take the role as verifier/claimant. The authentication preferences set by the application is used to determine in which direction the authentication takes place. Furthermore, some applications require only one-way authentication, while others, like peer-to-peer applications often use two-way authentication. Two-way authentication can be performed by executing the described steps twice. In the first round, device A may be the verifier and device B the claimant, while in the second round, the role is switched, such that B is the verifier and A is the claimant.

[3] Bluetooth Specification Version 4.1

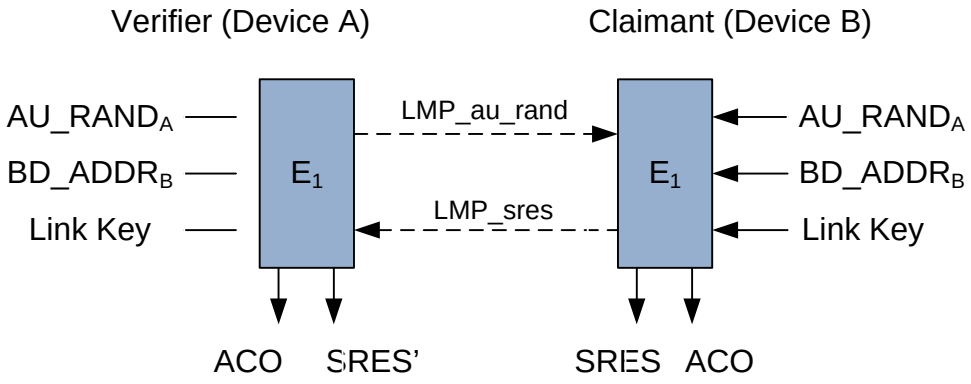

Figure 3.4 Legacy authentication procedure

3.3 Encryption

Encryption is the means to provide confidentiality on traffic exchanged over an unsafe communication channel. Bluetooth defines three settings for encryption as follows:

- *No encryption*, which is the default setting. In this setting no messages are encrypted.
- *Point-to-point encryption* enables encryption on unicast traffic. Broadcast traffic remains unencrypted. This setting can be enabled either during the connection establishment procedure or after the connection has been established.
- *Point-to-point and broadcast encryption* enables all messages to be encrypted. This setting can only be enabled after the connection between two parties has been established. This setting should not be enabled unless all affected links share the same master link key K_{master} as well as a common random value EN_RAND. Both of these parameters are used in the generation of the encryption key.

When it is desirable to enable encryption, the procedure shown in Figure 3.5 illustrates how this can be done. Broadly the steps shown may be grouped into three phases:

1. Initial setup
2. Generation of encryption key
3. Data encryption

The following subsections elaborate these phases.

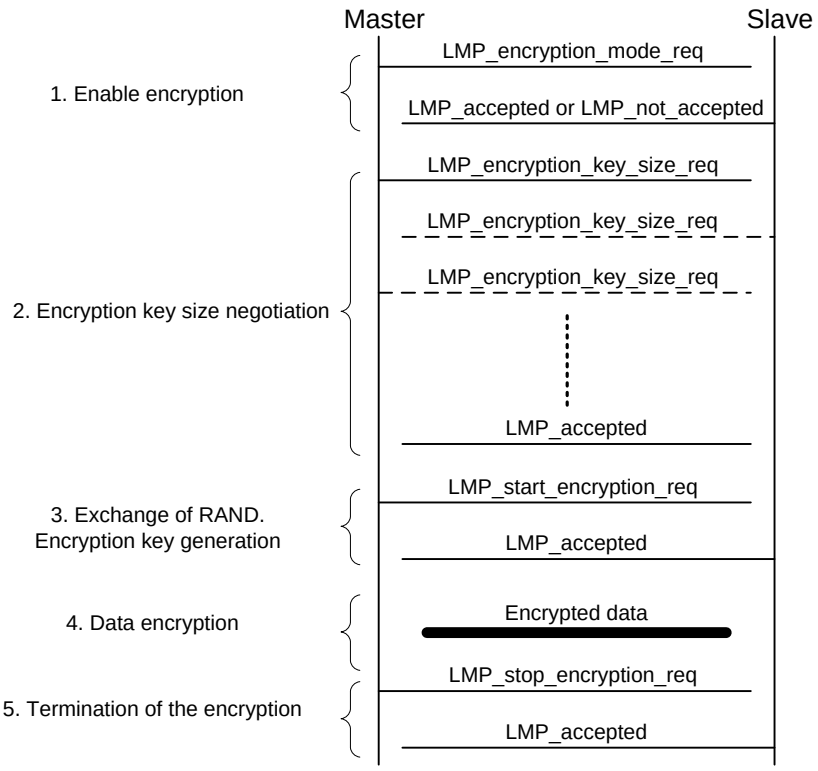

Figure 3.5 Activation and deactivation of data encryption

3.3.1 Initial Setup

Initially in step 1, the master and its slaves must agree upon whether to enable encryption. This is done by either party sending an LMP_encryption_mode_req message to suggest the activation of encryption. Upon receiving the message, the receiver replies with an LMP_accept message if it agrees to enable encryption; otherwise an LMP_not_accepted message is sent.

In step 2, the master and the slave(s) negotiate for the encryption key size to be used, which can be between 8-128 bits. The master initially sends an LMP_encryption_key_size_req message suggesting a key size L. If the slave supports the suggested key size, it will reply with an LMP_accepted message, otherwise it can send a new LMP_encryption_key_size_req in order to suggest an alternative key size L'. When the parties have come to an agreement, the negotiation is completed with an LMP_accepted message.

3.3.2 Encryption Key Generation

In step 3, the master generates a random number EN_RAND and sends it as plain text in the LMP_start_encryption_req to the slave(s). When receiving the message, the slave calculates the encryption key K_C and replies with an LMP_accepted message. When encryption is enabled on unicast /broadcast traffic, it is important that all slaves in the piconet receive the same random number so that the slaves can derive the same encryption key.

The calculation of K_C is performed using the E_3 function:

$$K_C = E_3(LK, EN_{RAND}, COF) \qquad (3.4)$$

where

$$COF = \begin{cases} BBD\_ADDR_A \cup BD_{ADDR_A}, & \text{if link key is a master key} \\ ACO & \text{otherwise} \end{cases}$$

The function above takes as input a 128-bit link key LK, a 128-bit random value EN_RAND and a 96-bit Ciphering Offset (COF). The link key may be a unit key K_A or a combination key K_{AB} in the case of unicast traffic encryption only. In the case when both broadcast traffic and unicast traffic need to be encrypted, the master link key K_{master} is used as the current link key and appended in Equation (3.4). The value of COF depends on whether the current link key is the master key or not. In the case when K_{master} is used, the COF value is a function of the master's address (i.e. BD_ADDR_A is concatenated twice to make a 96-bit value). Otherwise, when the current link key is K_A or K_{AB}, COF is equal to the ACO value generated during authentication.

3.3.3 Data Encryption

Once the encryption key is created, it is possible to encrypt/decrypt outgoing/incoming data (step 4). The process of data encryption/decryption is illustrated in Figure 3.6. The stream cipher E_0 function produces a keystream that is XOR'ed with the outgoing plain text data stream. The result is a ciphertext stream. On the receiver side, the same key stream is produced and XOR'ed with the incoming ciphertext in order to decipher it. Note that only the payload is encrypted, while the access code and the packet header remain unencrypted, as shown in Figure 3.7.

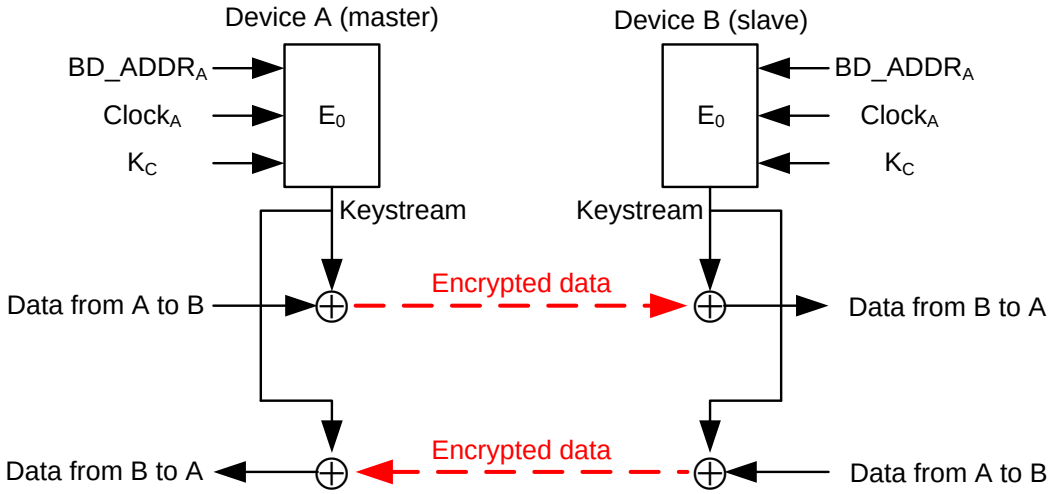

Figure 3.6 Encryption and decryption of data

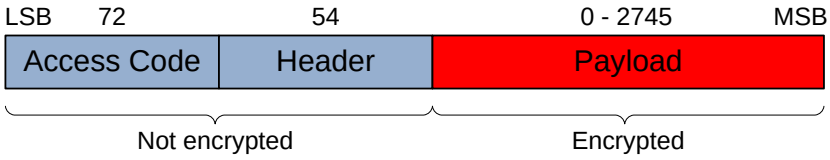

Figure 3.7 Example of an encrypted Bluetooth packet

Once encryption is not needed anymore, the master or slave can send LMP_stop_encryption_req message to notify the other device (step 5). Upon receiving the message, the receiver can accept the request by replying with an LMP_accepted message and then terminates encryption. Alternatively, the receiver may return an LMP_not_accepted message if it does not accept the request.

The E_0 function is derived from Massey-Rueppel algorithm[4] and takes as input the device address of the master BD_ADDR_A, 26 bits of the master real-time clock $Clock_A$, and the encryption key K_C. Note that the E_0 function is reinitialized at the start of each new packet to be sent or received. Effectively, this means that the clock value feeded into E_0 is updated for each new packet.

An important thing to be aware of is that the E_0 function is not a Federal Information Processing Standards (FIPS) approved algorithm and has come under scrutiny recently in terms of algorithmic strength (Padgette, Scarfone and Chen, 2012). Furthermore, the E_0 function does not provide any cryptographic message integrity protection. Even though the Cyclic Redundancy Check (CRC) of the payload provides some integrity protection, it is not considered cryptographically safe as it can be easily forged.

4 Secure Simple Pairing

Secure Simple Pairing (SSP) was introduced since Bluetooth version 2.1 and is mandatory for any device that implements this or later versions. The motivation of SSP is to simplify and improve the pairing procedure for the user and enhance the security in Bluetooth. The goal of SSP is to provide protection against passive eavesdropping and man-in-the-middle (MiTM) attacks, also called active eavesdropping. Compared with legacy security, SSP introduces some major changes:

- Elliptic Curves Diffie-Hellman (ECDH) Public key cryptography is used during link key establishment. ECDH is approved by FIPS.
- Introduction of association models to simplify the pairing process from the user point of view.

[4] Bluetooth Specification Version 4.1, Vol 1, page 89

- The link key in legacy pairing is derived from the PIN-code, the only source of entropy, and is therefore vulnerable to exhaustive search attacks. With SSP, link key generation is completely independent of the PIN-code. The entropy is much higher and is therefore much more secure.

4.1 Association Models

SSP defines four different association models (or pairing methods) to accommodate device diversity in terms of Input and Output (IO) capabilities. This implies that the association model used during the pairing process is dependent on the IO capabilities of the connecting devices, i.e. whether they have a display/numeric keyboard or not.

Numeric Comparison is designed for the case in which both Bluetooth devices can display a six-digit number and allow the user to enter "Yes" or "No" in response. During pairing, a six-digit PIN is shown on each display, and the user must answer "Yes" on each device if the PIN matches. Otherwise, the user must answer "No" such that the pairing process is terminated. An important difference in this process compared to Legacy Pairing is that the six-digit PIN is not used as input in the subsequent link key generation. The reason is to prevent an attacker who oversees the displayed number from determining the resulting link or encryption key. This association model provides to a certain degree protection against MiTM attacks. The chance to crack is only 1 in a million during a pairing trial, and assuming a 6 digits PIN[5]. Even though there is a theoretical small chance for MiTM attack, it is anticipated that in practice, performing this type of attack is extremely difficult due to very short distance between the target devices in most use cases. The numeric comparison model has been proved to be secure (Lindell, 2009).

Passkey Entry is designed for the case where one Bluetooth device has a numeric keyboard while the other device has a display but no keyboard. During pairing, a six-digit PIN is shown on the device with a display, and the user must enter this PIN on the second device with the keyboard. As with Numeric Comparison, the six-digit PIN is not used as input in the subsequent link key generation. Passkey Entry is designed to provide protection against MiTM attaks. However, it has been shown that due to design weakness, it is possible to calculate the PIN, if an attacker can record the first part of the pairing process. This enables the attacker to mount a MiTM attack. A more detailed description on this attack is provided in section 6.2.

Just Works is designed for the case where one (or both) of the pairing devices has neither a display nor a keyboard, e.g. a Bluetooth headset. The pairing procedure is the same as in the Numeric Comparison except that the six-digit PIN is not displayed and the user is neither asked for confirmation. In this model, the user has to accept a connection without verifying the six-digit PIN. Therefore, this association model may be susceptible for MiTM attacks.

Out of Band (OOB) is designed for devices that support an alternative and secondary wireless technology in addition to Bluetooth. Near Field Communication (NFC) is an example. With the

[5] Bluetooth Specification Version 4.1, Vol 2, page 1326

OOB association model, device discovery as well as exchange of cryptographic parameters during pairing is performed through this alternative wireless channel. The assumption here is that the secondary communication channel is secure for exchange of cryptographic parameters, and hence allows the pairing process to be performed securely. This will enhance user experience and ease the process of pairing devices. As an example, with NFC, the user just has to touch two devices together to initiate the pairing process. The user will then be asked to accept the pairing which can be accomplished with a single button push.

4.2 Phases in SSP

SSP consists of 6 phases as shown in Figure 4.1. Note that all phases, except phase 3, are executed in the same way independent of the association model. Only in phase 3, i.e. authentication stage 1, the procedure is different depending on which association model is used.

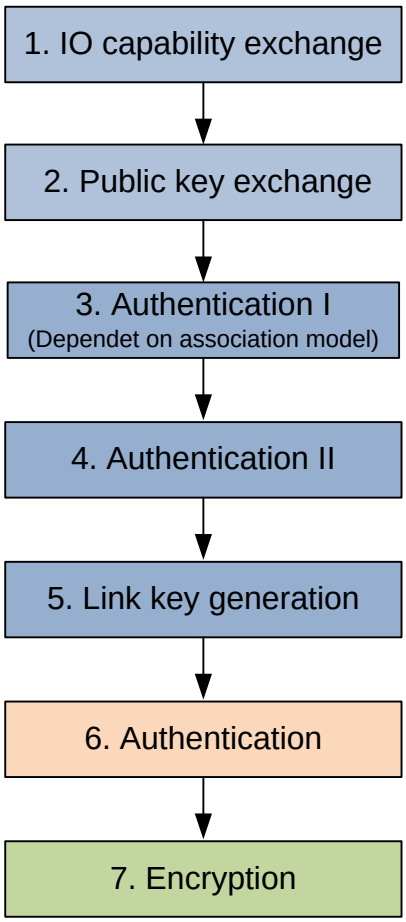

Figure 4.1 Phases in Secure Simple Pairing

Phase 1 – IO capability exchange
Initially the Input/Output capability of the two devices is exchanged in order to determine the appropriate association model to be used.

Phase 2 – Puclic key exchange

Next, the ECDH public key is exchanged. Initially in step 1, each device generates its own ECDH public-private key pair (PKx and SKx). The public key (PKa and PKb) is then exchanged between the two devices in steps 2 and 3. SSP uses P-192 ECDH, meaning that the prime number *p* of the elliptic curve algorithm is 192-bits long.

Once the public keys are exchanged, both parties, i.e. the initiating and the responding devices, commence to calculate the shared secret DHKey (step 4), which is simply a point product of the local private key and the remote public key, i.e. DHKey = SKa·PKb = SKb·PKa. This process may however take some times, so meanwhile, the procedure in phase 3 may be executed.

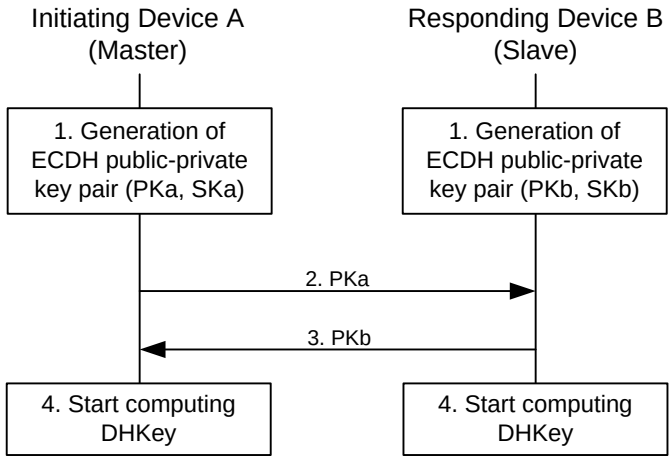

Figure 4.2 Public key exchange

Phase 3 – Authentication Stage 1

The purpose of this phase is to perform an initial authentication using PIN code. The procedure used is dependent on the applied association model. Each of the following association models: Numeric Comparison, Out-of-band and Passkey Entry, have a specific authentication procedure. No separate authentication procedure is defined for the Just Work association model, as this model can be regarded as a derivation of the Numeric Comparison model and therefore may use the same procedure.

For simplicity we will only describe the authentication procedure for the case of Numeric Comparison (see Figure 4.3) as this is assumed to be the most commonly used association model (e.g. used between smart phones, laptops, etc.). The authentication procedures for the Out-of-band and Passkey Entry association models are in principal similar to the procedure of Numeric Comparison, but with some modifications. For interested readers, we refer to the specification for further details (Bluetooth Specification, version 4.1, 2013).

Initially in step 1, both the initiating and responding device select a pseudo-random number, Na and Nb, respectively. The random numbers must be freshly generated each time the steps in this phase are performed in order to prevent replay attack.

Next, in step 2, the responding device computes a value called commitment Cb, using the two public keys from phase 1, i.e. PKa and PKb, and the random number Nb, as input parameters. The commitment Cb is then sent over to the initiating device in step 3.

In step 4 and 5, the random numbers Na and Nb is exchanged.

Next, in step 6, the initiating device computes a Cb value and checks if this value is equal to the Cb received in step 3. A failure in this check either indicates the presence of an attacker or other transmission errors. If the check fails, all further execution of the authentication procedure is aborted, and the procedure must be restarted from the beginning.

Note that the steps in 3-6, may be regarded as a reverse challenge-and-response scheme, where the responding device first sends Cb (the answer) before the random values (the challenge), in which the initiating device uses to calculate the same Cb, are exchanged.

In step 7, both sides compute a 6-digit PIN (Va and Vb) that is displayed to the user on their devices. The calculation is based on the random values Na and Nb and the public keys PKa and PKb. The user must then check and confirm that these values match or not (step 8).

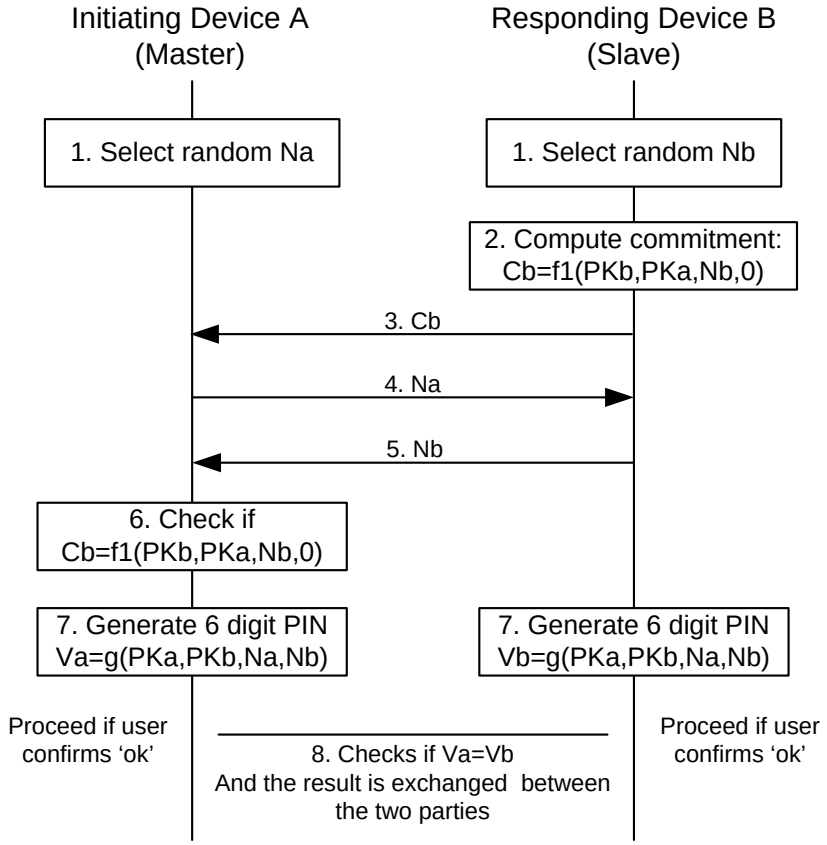

Figure 4.3 Authentication stage 1

Phase 4 – Authentication Stage 2

Before executing the steps in the second authentication stage as shown in Figure 4.4, it is expected that both devices have finished calculating the shared secret key, DHKey from phase 2. The main purpose of this stage is to verify that this key is identical on both sides. In order to fulfil this verification, both the initiating and the responding device calculate a confirmation value Ea and Eb (step 1), using function f3. The input parameters to this function include among others: DHKey, the IO capability information exchanged in phase 1 of 3 octets length, the BD_ADDR of device A and B, and random values. The complete list of input parameters is shown in Figure 4.4, and explained in detail in Table 4.1.

Next, in step 2, A sends its confirmation value Ea to B. Device B computes an equivalent Ea value and checks if the computed value matches the received value (step 3). If this check fails, further execution is aborted. If the check is successful, B sends its Eb value to A in step 4. Upon receiving Eb from B (step 4), device A performs the same check as in step 3.

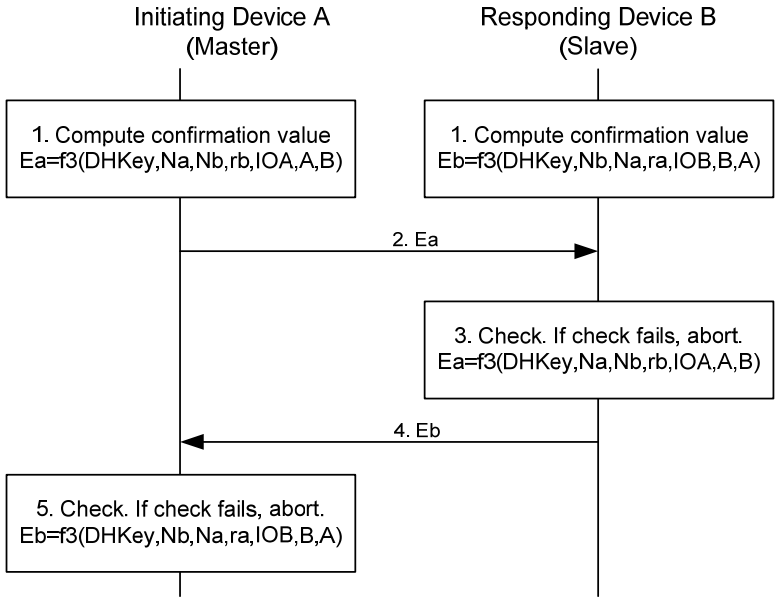

Figure 4.4 Authentication stage 2

Table 4.1 Input parameters to f3 function to generate a confirmation value

Input Parameter	Description
DHKey	Shared secret Diffie Hellman key
Na/Nb	Random nonce generated by node A/B
ra/rb	Random value generated by node A/B. This value is used only in OOB and Passkey Entry association model. In the case of Numeric Comparison ra=rb=0.
IOA/IOB	IO-capability of node A/B
A/B	BD_ADDR of node A/B

Phase 5 – Link key calculation

In phase 5, both parties compute the link key LK using the function f2 as given in Equation (4.1). The input parameters to this function include the shared secret key DHKey, the fixed string "btlk", and other publicly exchanged data. The notations for these input parameters are already described in Table 4.1.

$$LK = f2(DHKey, Na, Nb, "btlk", A,B) \qquad (4.1)$$

An important difference between this SSP link key and the legacy pairing link key in Section 3.1, is that the SSP link key is generated entirely independent of the passkey (equivalent to PIN). This difference allows for a stronger link key, where the strength is given by the amount of entropy (randomness) in the key generation process. For SSP, the entropy is approximately 96 bits when FIPS-192 elliptic curve is used, and 128 bits when using FIPS-256 elliptic curve[6]. On the other hand, in the case of legacy pairing, the only source of entropy is the PIN which, in many use cases, is typically four digits (or about 13 bits entropy), either selected by the user or fixed for a given product. Due to this low amount of entropy, it is possible to run a record attack (on the pairing procedure and one authentication exchange) in order to find the PIN in a very short amount of time as pointed out earlier.

Phase 6 and 7 –Authentication and Encryption

The final phases 6 and 7 in SSP are performing authentication, generating the encryption keys and performing data encryption. These procedures are the same as the final steps in legacy pairing described in Section 3.

5 Secure Connections

Version 4.1 of the Bluetooth Specification introduced Secure Connections, which is actually an upgrade of Secure Simple Pairing (SSP). The only difference between SSP and Secure Connections, is that the latter uses even stronger and more secure algorithm approved by FIPS as shown previously in Table 2.1, on the following procedures: key generation, authentication and encryption. The characteristics of Secure Connections are summarized below:

- P-256 ECDH is used for key generation
- HMAC-SHA-256 is used for authentication. In contrast to legacy authentication, the authentication procedure in Secure Connections, referred to as Secure Authentication, is by nature a two-way authentication scheme.
- AES-CCM is used for encryption. The advantage of AES-CCM is that it provides both message confidentiality and integrity protection.

[6] Bluetooth Specification, Version 4.1, Vol 1, page 91

The use of AES-CCM for encryption provides not only confidentiality protection but also integrity protection. As stated earlier, only Secure Connections provides integrity protection with cryptographic strength. Legacy pairing and SSP, on the other hand, only provide CRC check, which cannot be regarded as integrity protection.

6 Know Threats and Attacks?

The Bluetooth technology has gained increased popularity and wider usage over the last decade. This has led to greater attention and interest from hacker communities. Being a wireless technology, it is inevitably susceptible from attacks such as eavesdropping, Man-in-The-Middle (MiTM), Denial-of-Services, (DoS), and jamming. In addition, as Bluetooth-enabled devices, such as smart phones and PDAs, can perform the same tasks as computers, they are also vulnerable for computer viruses (Loo, 2009).

A number of security breaches and threats have been discovered over the years. The first Bluetooth hacking threat was discovered in 2003 with the release of Bluesnarfing (Dunning, 2011). Following this discovery, a number of other security breaches have been discovered. These security breaches and threats are basically caused by several reasons:

- Weaknesses in the Bluetooth specification, especially in the earlier Bluetooth versions.
- Side-effects of design features
- Improper implementation by manufacturers
- Improper use by the user

The literature provides a wide variety of threats to the Bluetooth system (Harte, 2009; Curt and Layton, 2014; Loo, 2009; Tan and Masagca, 2011), of which we briefly describe in the following subsection

6.1 A Variety of Threats

Bluesnarfing

Bluesnarfing enables an attacker to anonymously connect to some Bluetooth-enabled mobile phone without the owner's consent and extract restricted data from the phone such as the address book, call log, SMS, and the calendar (Legg, 2005; Laurie, 2003; Herfurt, 2004).

What makes this attack possible is the loophole in the OBEX[7] Push profile, which allows a connection to be established without authentication, as this profile was designed to send non-malicious data such as electronic business cards (vCard). It was deemed unnecessary to require authentication for such simple exchanges by the mobile phone engineers. Once a connection to a target phone is established, the "pull" function is executed, instead of the "push" function, to retrieve data from the target device.

[7] OBEX stands for Object Exchange Protocol, and is a protocol which facilitates exchange of data objects

Bluesnarfing attacks basically work on visible or discoverable phones, but according to (Legg, 2005) the same attack can be performed on non-discoverable devices as well. Bluesnarfing needs only to know the Bluetooth device address of the target device, which can be determined using third party software. One example is RedFang written by Ollie Whitehouse which is basically a brute-force approach. The program iterates through all possible combination of the 6 bytes long address in order to search for the correct one. The main difficulty in such an approach is that the process can take hours of computing time. However, with tools like the Ubertooth[8], this process can be shortened down drastically if the target device is active, i.e. transmits traffic, and thus reveals its existence.

At the CeBIT IT-exposition in 2004, a field trial was conducted on the fairground which confirmed that some mobile phone models from leading companies at that time such as Nokia, SonyEricsson and Ericsson, were vulnerable to the bluesnarfing attack. The trial demonstrated that it is possible to retrieve phone books from the attacked phones. As the security flaw became publicly known, new firmware versions have been released to fix the issue. It is unclear whether this attack is possible today with newer phones or not. Most likely it is outdated. However, this attack is a good example of how a simple and innocent design feature can have unexpected side-effects with disastrous consequences.

Bluejacking

Like bluesnarfing, bluejacking also exploits the same loophole in OBEX Push profile to send unsolicited short messages to the target device. This can be done by sending a fictitious business card where the message e.g. “Hello World” is inserted in the name field of the business card. Bluejacking is mostly performed for amusement and is usually harmless as no personal data is retrieved. However, bluejacked people generally don't understand why such messages appear on their phone, and may think that the phone is malfunctioning. Besides, bluejacking may be used in guerrilla marketing campaigns and advertisements which can be very annoying. The bluejacking attack is probably obosolete.

Bluebugging

Bluebugging is a powerful and dangerous attack that allows the attacker to take control over the victim’s mobile phone (Legg, 2005). This enable the attacker to perform a number of tasks without the owner’s knowledge, such as initiating calls, send SMS, activate call forwarding and thereby receive calls intended for the victim. In addition bluebugging also has all the same capability as bluesnarfing, i.e. to retrieve restricted data such as the phone book, and also to alter the data. Only a few early Bluetooth enabled mobile phones were vulnerable to this attack, due to faulty and improper implementations of the Bluetooth protocols. In order to perform bluebugging, the attacker must somehow pair his phone with the target device. According to a security consultant, Ollie Whitehouse, the creator of the Redfang tool, this can be done by sniffing the data traffic during a pairing with another device, for instance a Bluetooth headset. With this information, the attacker can establish a connection with the target device by pretending to be the trusted device (Newitz, 2004). Once a connection is established, the attacker can send AT

[8] http://ubertooth.blogspot.no/

commands in order to instruct the bluebugged phone to perform any desired tasks such as initiating a call. AT commands is an abbreviation for *attention commands*, also known as the Hayes command set, which is a command language commonly used in computer telecommunication. It is likely that this attack is obsolete.

PIN-cracking

As previously mentioned in section 3, the legacy pairing procedure is inherently insecure and susceptible for being cracked, due to the low entropy of the PIN. The work in (Shaked and Wool, 2005) shows that the PIN and consequently the link key can be easily cracked or determined. Given that the whole sequence of the pairing and the authentication procedures have been eavesdropped and recorded, the attacker can use a brute force method to search for the correct combination of the PIN. This attack can be performed entirely passively (no packet injection) without the target's awareness. The searching time for the correct PIN depends on the length of the PIN. However, very often, the PIN is only 4 digits long. With a Pentium IV 3 GHz computer, a 4-digits PIN can be cracked in less than 0.06 sec, while a 7-digits PIN takes 76 seconds to crack. A disadvantage of this attack is that it requires the sequence of pairing and authentication to be captured. However, in reality, these procedures are very seldom executed. Most people pairs their devices in non-public spaces, and once a pair of devices have been paired, no further pairing is necessary in later connection establishments. Thus, to be able perform the PIN-attack, an attacker needs also to force the user to re-pair their devices. This attack is known as the re-pairing attack, and is discussed below. The PIN-cracking attack is probably applicable on older Bluetooth devices which use Legacy Security, but not possible on newer devices which support SSP/Secure Connections.

Re-pairing attack

In order to make the PIN-attack realizable, (Shaked and Wool, 2005) suggests several concepts for performing the re-pairing attack, in which the aim is to trick the user to re-perform the pairing procedure. This is possible as the Bluetooth specification does allow devices to forget or lose their link keys. When this happens, a re-pairing procedure can be performed in order to establish a new link key. Thus, to perform the re-pairing attack, the attacker must manipulate the devices to "think" that the link key has been lost by actively injecting forged packets during the authentication procedure. Recall that during the authentication the master sends the random value AU_RAND, which is the "challenge" in which the slave must respond with a correct SRES value, which is the answer. However, an attacker may trick the master to believe that the link key has been lost by injecting an *LMP_not_accepted* message to the master. The master will then discard the stored link key and initiate re-pairing. Alternatively, before the master sends the AU_RAND, the attacker can inject an IN_RAND message to the slave and manipulate it to think that the master has lost the link key, such that a re-pairing is performed.

Virus infection

Viruses on Bluetooth were first introduced in 2004. Early cell-phone virus writers have taken advantage of Bluetooth's automated connection process to send out infected files. However, since most cell phones use a secure Bluetooth connection that requires authorization and authentication

before accepting data from an unknown device, the infected file typically doesn't get very far. When the virus arrives in the user's cell phone, the user has to agree to open it and then agree to install it. This has, so far, stopped most cell-phone viruses from doing much damage.

Denial of service (DoS)

DoS attacks are also possible on Bluetooth enabled devices. This works in the same way as the traditional DoS attacks in which the attacker continuously sends invalid request messages in order to keep the Bluetooth channel busy. DoS attack will not only block the usage of any Bluetooth service but also drain the battery of the target device (S. Ho, B. Ng, J. Kwong, F. Wu).

Blueprinting

Blueprinting is a surveillance attack in which it allows the attacker to discover the fingerprint of Bluetooth-enabled devices. The information that can be retrieved comprises details uniquely identifying a particular device, i.e. producer, model, and unique address of the equipment. This information can be used to find devices that are susceptible for attacks.

A threat taxonomy that serves as a framework for classifying Bluetooth-based threats is given is Table 6.1 (Dunning, 2011). The taxonomy provides nine distinct classes; surveillance, range extension, obfuscation, fuzzer, sniffing, DoS, malware, unauthorized direct data access and Man-in-The-Middle (MiTM) attacks. The attack classes are also categorized according to threat level;

Table 6.1 Bluetooth threat taxonomy (Dunning, 2011)

Threat	Tool	Consequences for the victim
Surveillance	Blueprinting, bt_audit, redfang, war-nibbling, Bluefish, sdptool, Bluescanner, BTScanner,	Low
Range extension	Bluesniping, bluetooone, Vera-NG	Low
Obfuscation	Bdaddr, hciconfig, Spooftooph	Low
Fuzzer	BluePass, Bluetooth Stack Smasher, BlueSmack, Tanya, BlueStab	Medium
Sniffing	FTS4BT, Merlin, Bluesniff, HCIDump, Wireshark, Kismet, Ubertooth	Medium
Denial of Service	Battery exhaustion, signal jamming, BlueSYN, Blueper, BlueJacking, vCardBlaster	Medium
Malware	BlueBag, Caribe, CommWarrior	Medium
Uautorized direct data access	Bloover, BlueBug, BlueSnarf, BlueSnarf++, BTCrack, Car Whisperer, HeloMoto, btpincrack	High
«Man-in-the-Middle»	BT-SSP-printer-MITM, BlueSpooof, bthidproxy	High

low, medium and high, where low means that the attack is generally harmless on its own, medium can cause inconvenience or be malicious, while high means that the victim can suffer a lot as the attacker gets access to the data. Note that the table contains a greater diversity of attacks that what have been described in this report, as it is beyond the scope of this report to give a complete description of them all. The important thing is that the table illustrates the diverse and numerous vulnerabilities in the Bluetooth standard. With the introduction of newer Bluetooth versions as well as Secure Simple Pairing, the security has in general matured, and many of these vulnerabilities are probably obsolete. However, older Bluetooth devices may still be vulnerable to these threats.

6.2 SSP Vulnerabilities

Even though the introduction of Secure Simple Pairing (SSP) has resulted in enhanced security, several publications have shown that there are still vulnerabilities in some of the association models of SSP.

According to (K. Haataja and P. Toivanen, 2010) during a SSP pairing procedure, devices must exchange their IO capabilities in order to agree upon an appropriate pairing mode. This exchange is performed over an unauthenticated and unsecure channel, and can therefore be exploited by an adversary. An attacker can modify the exchanged IO capabilities information in order to manipulate the target devices to use a desired weaker association model. Usually, the preferred association model is Just Works as this model does not provide any protection against MiTM attacks.

Furthermore, it is shown that the Passkey Entry model of SSP is vulnerable against a MiTM attack (J. Barnickel, J. Wang and U. Meyer, 2013). Recall that the PIN used in SSP is composed of a 6 digits number. During Authentication I in Figure 4.1, this PIN is converted to a 20 bits string equivalent in which it is verified bit by bit, in 20 rounds. Because of design weakness, i.e. the bitwise verification, the attacker can easily calculate the 20 bits equivalent of the PIN once the entire message exchange during this authentication procedure is recorded. The attacker can next jam further progression of the pairing process in order to force a new pairing. In the second pairing try, if the same PIN is reused, the attacker can then use his or her knowledge about the PIN to perform MiTM attack. The prerequisite for this attack to work is that the same PIN is reused on the second try. This is naturally the case when one of the devices uses a fixed PIN. Even if the PIN is chosen and manually entered by the user, it may be very plausible that the same PIN is used again.

The Just Works model does not provide any protection against MiTM attacks as stated earlier, and is thus inherently vulnerable. On the other hand the Numeric Comparison model has been proven to be secure (Y. Lindell, 2009).

7 How to Mitigate Threats?

Although the security has improved in later versions of the Bluetooth specification, there are still weaknesses in the security architectures, as discussed in the previous chapter. One of the main challenges in designing security lies in the fundamental contradiction of security and user friendliness. It is in general very difficult to design a secure system that at the same time is user friendly. Often a trade-off is done between these two factors. This is also the case in Bluetooth. Therefore, when using Bluetooth, it is advised to follow a number of measures in order to mitigate the risk for being attacked:

- When Bluetooth is enabled, make sure it is in "hidden" or "non-discoverable" mode most of the time, except when inquiring and pairing occurs. The hidden mode prevents other Bluetooth devices from recognizing your device. This does not prevent the user from using Bluetooth.
- Be careful where to use Bluetooth and be aware of the surrounding environment. This is especially important when performing pairing or operating in discoverable mode. For example, if using Bluetooth in public places, there is a greater risk that someone else may be able to intercept the connection than if using it at home.
- Do not accept any unsolicited message, and do not accept inquiry or pairing requests from strangers.
- Also be aware that the Bluetooth range can be greater than anticipated. Hackers have used directional high-gain antenna to successfully communicate over much greater distances. A good example is the BlueSniper Rifle which can increase the Bluetooth radio range to over 1.5 km[9].
- Take advantage of security options - Learn what security options your Bluetooth device offers, and take advantage of features like authentication and encryption.
- Keep the Bluetooth software updated to the newest version. Older Bluetooth devices use versions of the Bluetooth protocol that have security holes. Even the latest specification (4.0) has a similar problem with its low-energy (LE) variant. To combat this threat: Ban devices that use Bluetooth 1.x, 2.0 or 4.0-LE.

[9] http://www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html

8 Conclusion

Bluetooth has since its invention gained increasingly higher popularity and is now a de-facto standard for short-range radio communication. The technology is applied in many areas such as consumer electronics, industry automation, wireless sensor networks (WSN), health and fitness, and automobile. An area in which it is expected to achieve exponential growth in coming years is related to Internet of Things (IoT). The proliferation and wide adoption of Bluetooth has stimulated to increased focus on the security of the technology. The motivation of this report is thus to provide an in-depth description of the security mechanisms in Bluetooth and its vulnerabilities.

Since Bluetooth communicates over the unsecure wireless medium, it is inevitably vulnerable to several threats, including eavesdropping, jamming, Man-in-The-Middle (MiTM) and Denial-of-Service (DoS) attacks. In order to provide protection against these potential threats, the Bluetooth standard supports several security mechanisms. In principle there are three main mechanisms i.e. authentication, confidentiality and message integrity. Additionally, there is a mechanism for link key establishment referred as pairing. This mechanism plays a crucial role and lays the foundation for performing the above security mechanisms. This implies that the strength of these mechanisms is dependent on the strength of the established link key. A weak pairing mechanism can result in the compromise of the link key, and consequently, the security mechanisms.

Over the years, a number of security breaches have been documented, ranging from harmless to very serious flaws. In the most harmful cases, it is possible to gain access to restricted data and even take the complete control over the victim’s Bluetooth-enabled device. Most of these security flaws are mainly caused by weak security architecture and/or improper implementations of the standard. Especially earlier Bluetooth versions (versions 2.0 and earlier) which used legacy security or legacy pairing were vulnerable due to weaknesses in the pairing mechanism. With the introduction of Secure Simple Pairing (SSP), the security in Bluetooth has improved and is considerably more secure. However, there are still weaknesses in SSP due to compromises between security and usability. From a user point of view, it is important to be aware of the potential risk and learn how to protect the Bluetooth device. There are a number of measures that the user can do to alleviate the risk for being attacked.

References

J. Barnickel, J. Wang and U. Meyer, “Implementing an Attack on Bluetooth 2.1+ Secure Simple Pairing in Passkey Entry Mode,” Bluetooth Specification, Version 4, 03. December 2013: 17.

Dunning, J.P., «Taming the Blue Beast: A Survey of Bluetooth Based Threats”, *Security & Privacy, IEEE,* 8 (2): 20-27.

Franklin, C, and J. Layton. "How Bluetooth Works" 28 June 2000. HowStuffWorks.com. 2014, http://electronics.howstuffworks.com/bluetooth.htm, downloaded 15th March 2014.

Hagen, J. M., “How do employees comply with security policy? A comparative case study of four organizations under the security act”, In: *The Human factor behind the Security Perimeter. Evaluating the effectiveness of organizational information security measures and employees’ contribution to security*, Doctoral dissertation, Faculty of Mathematics and Natural Sciences, University of Oslo, 2009.

K. Haataja and P. Toivanen, “Two Practical Man-In-The-Middle Attacks on Bluetooth Secure Simple Pairing and Countermeasures,” IEEE Transactions on Wireless Communications, Vol. 9, No. 1, 2010.

K. Haataja, “*Security Threats and Countermeasures in Bluetooth-Enabled Systems, Doctoral dissertation”*, Department of Computer Science, The Faculty of Business and Information Technology of the University of Kuopio, Friday 6th February 2009, Available at: http://epublications.uef.fi/pub/urn_isbn_978-951-27-0111-7/urn_isbn_978-951-27-0111-7.pdf, downloaded 7th April 2014.

K. Haataja, K. Hyppönen, S. Pasanen, P. Toivanen, “*Blueetooth Security Attacks, Comparative Analysis, Attacks, and Countermeasures,*” Springer Heldelberg, New York Dordrecht London, 2013.

Harte, L., “*Introduction to Bluetooth. Technology, Market, Operation, Profiles, and Services”*, 2nd Edition. Althos Publishing, Fquay – Varina, USA.

Herfurt, M., “Bluesnarfing @ CeBit 2004 - Detecting and Attacking Bluetooth-enabled Cellphones at the Hannover Fairground”, http://trifinite.org/Downloads/BlueSnarf_CeBIT2004.pdf, 2004

Massey, J., Khachatrian, G., Kuregian, M., “Nomination of SAFER+ as Candidate Algorithm for the Advanced Encryption Standard (AES)”, AES Candidate Conference, 1998

Jennings, J., Webroot, A Review of Bluetooth Attacks and How to Secure Mobile Workforce Devices, http://www.webroot.com/us/en/business/resources/articles/corporate-security/a-review-of-bluetooth-attacks-and-how-to-secure-mobile-workforce-devices, downloaded 26th March 2014.

Laurie, A., “Serious flaws in Bluetooth security lead to disclosure of personal data”, http://www.iwar.org.uk/news-archive/2003/11-13-12.htm, 2003,

Legg, G., “The Bluejacking, Bluesnarfing, Bluebugging Blues: Bluetooth Faces Perception of Vulnerability”, http://www.eetimes.com/document.asp?doc_id=1275730, 2005

Lindell, Y., “Comparison-based key exchange and the security of the numeric comparison mode in bluetooth v2.1”. In The Cryptographers’ Track at the RSA Conference, 2009

Loo, A., “Security Threats of Smart Phones and Bluetooth”, *Communications of the ACM*, March 2009, Vol 52. No 3: 150-152. DOI: 10.1145/1467247.1467282

Minar, N. B. I. and Mohammed Tarique, M., “Bluetooth Security Threats and Solutions, A Survey”, *International Journal of Distributed and Parallel Systems* (IJDPS) Vol.3, No.1, January 2012.

Mindi McDowell and Matt Lytle , “Security Tip (ST05-015), Understanding Bluetooth Technology”, Article, Original release date: September 08, 2010, Last revised: February 06, 2013, https://www.us-cert.gov/ncas/tips/ST05-015, downloaded 15.03.14.

Newitz, A., “*They’ve Got Your Number* …”, Wired (12/04) Vol. 12, No. 12, P. 92, 2004

Nilsson, Rolf, “*Technology Update: Connectivity of things:Wireless for the last 100 m of IoT,*” *Control Engineering*, March 2014: 16.

Padgette, John, Scarfone Karen and Lily Chen, “*Guide to Bluetooth security, Recommendations of the National Institute of Standards and Technology”*, NIST Special Publication 800-121 Revision 1, June 2012, National Institute for Standards and Technology.

Padgette, J.D., “Bluetooth security in the DoD”, *Military Communications Conference, 2009. MILCOM 2009. IEEE* 2009: 1 – 6. DOI: 10.1109/MILCOM.2009.5379967

Schneier, B., *Secrets and Lies: Digital Security in a Networked World, with New information about Post-9/11 Security,* 2nd Ed., Wiley Publishing, 2004.

Sandhya, S. and Devi, K.A.S., “Analysis of Bluetooth threats and v4.0 security features”, *International Conference on Computing, Communication and Applications (ICCCA),* 2012: 1 - 4 DOI: 10.1109/ICCCA.2012.6179149.

Shaked, Y., and Wool, A., “*Cracking the Bluetooth PIN*”, MobiSys ’05: The Third International Conference on Mobile Systems, Applications, and Services (MobiSys), 2005

Spill, D and Bittau, A., "*Bluesniff: Eve meets Alice and Bluetooth*", WOOT '07 Proceedings of the first USENIX workshop on Offensive Technologies, Article no 5, 2007.

Tan, M. and K.A. Masagca, , "An Investigation of Bluetooth Security Threats", *International Conference on Information Science and Applications (ICISA)*, 2011: 1 – 7, DOI: 10.1109/ICISA.2011.5772388.

Ablon, L., Libicki, M.C., and A.A. Golay, *Markets for Cybercrime Tools and Stolen Data Hackers' Bazaar*" , Research report, Rand Corporation, 2014, available at: http://www.rand.org/content/dam/rand/pubs/research_reports/RR600/RR610/RAND_RR610.pdf, downloaded 29.10.2014.

S. Ho, B. Ng, J. Kwong, and F. Wu "Security Analysis of Bluetooth Enabled Mobile Devices" http://www.bluetooth.com

Abbreviations

ACO	Authentication Ciphering Offset
AES-CCM	Advanced Encryption Standard Counter with CBC-MAC
AES-CTR	Advanced Encryption Standard Counter Mode
AMP	Alternate MAC/PHY
AT	Attention Commands
BR	Basic Rate
BYOD	Bringing Your Own Device
CBC	Cipher Block Chaining
COF	Ciphering Offset
CRC	Cyclic Redundancy Check
DoS	Denial of Service
ECDH	Elliptic Curve Diffie Hellman
EDR	Enhanced Data Rate
FHSS	Frequency Hopping Spread Spectrum
FIPS	Federal Information Processing Standards
HMAC	Keyed-Hash Message Authentication Code
IO	Input and Output
IoT	Internet of Things
L2CAP	Logical Link Control and Adaption
LE	Low Energy
LMP	Link Manager Protocol
MAC	Message Authentication Code
MiTM	Man in The Middle
NFC	Near Field Communication
OBEX	Object Exchange Protocol
OOB	Out of Band
PDA	Personal Digital Assistant
PIN	Personal Identification Number
SAFER+	Secure And Fast Encryption Routine+
SCADA	Supervisory Control and Data Acquisition
SHA	Secure Hash Algorithm
SMS	Short Message Service
SRES	Signed Response
SSP	Secure Simple Pairing
WSN	Wireless Sensor Networks